FORSCHUNGSZENTRUM JÜLICH GmbH
Jülich Supercomputing Centre
D-52425 Jülich, Tel. (02461) 61-6402

Interner Bericht

Entwicklung eines rollenbasierten Reporting-Systems am Beispiel des Jülich Supercomputing Centre

Cornelia Geyer

FZJ-JSC-IB-2009-02

Mai 2009

(letzte Änderung: 20.05.2008)

Kurzfassung

Das Jülich Supercomputing Centre (JSC) betreibt zentrale Supercomputer- und Server-Systeme und ist eines der leistungsstärksten wissenschaftlichen Rechenzentren in Europa. Als solches stellt es Wissenschaftlern am Forschungszentrum Jülich, an Universitäten und Forschungseinrichtungen in Deutschland und z.T. in Europa sowie der Industrie Rechenkapazität der höchsten Leistungsklasse zur Verfügung.

Das Reporting im JSC gibt Aufschluss über den Betrieb und die genutzten Ressourcen der Rechnersysteme. Dabei stellen die Berichte die notwendigen Daten auch für zukünftige Entscheidungen z.B. zur Planung der Rechenzeitverteilung bereit. Bisher wurden für verschiedene Anforderungen unterschiedliche Reports erzeugt. Einerseits existierte für die jeweiligen Benutzergruppen kein einheitlicher Zugang zu den Daten, andererseits wurden dazu jeweils andere Autorisierungsmechanismen benutzt. Darüberhinaus wurden nur vordefinierte Reports angeboten und es gab keine Möglichkeit konfigurierbare Reports automatisch zu erzeugen. Die grafische Ausgabe beschränkte sich lediglich auf die vom Administrator erstellten Monats- und Jahresauswertungen.

Im Rahmen dieser Diplomarbeit wurde ein Konzept für ein rollenbasiertes Reporting-System entwickelt, das diese Schwachstellen beseitigt. Dazu wurde zunächst die derzeitige Vorgehensweise des Reportings analysiert und die Anforderungen und Ziele an das neue System definiert. Für die drei Kernpunkte: einheitlicher und sicherer Zugang, Bereitstellung der Daten und deren Darstellung wurden geeignete Lösungen zur Umsetzung erarbeitet. Schließlich wurde das entwickelte Reporting-System in einer konkreten Implementierung realisiert und getestet.

Abstract

The Jülich Supercomputing Centre (JSC) operates central supercomputers and server systems. It is one of the most powerful scientific computer centres in Europe. As such it provides supercomputer resources of the highest performance class for scientists at the Research Centre Jülich, at universities and research institutions in Germany and partially in Europe as well as for industry.

Reporting at the JSC informs about the usage of the supercomputer systems. The reports also provide the necessary data for future decisions, e.g. for the planning of the compute time distribution. Up to now, different reports were created for different requirements. On the one hand, there was no uniform access to the data for different user groups, on the other hand, for each case other authorisation mechanisms were used. Furthermore only pre-defined reports were available and there are no possibilities to automatically create configurable reports. The graphical charts were offered only on a monthly or yearly basis pre-generated by the administrator of the reporting system.

In the context of this diploma thesis a concept was developed for a role-based reporting system which eliminates these weak points. Therefore the current procedure was analysed and the requirements and goals were defined. For the three issues: uniform and secure access, provision of the data, and their representation different solutions for realisation were developed. Finally, the new designed reporting system was implemented and tested.

Inhaltsverzeichnis

Abbildungsverzeichnis

Abkürzungsverzeichnis

API	Application Programming Interface
BI	Business Intelligence
BIRT	Business Intelligence and Reporting Systems
CA	Certification Authority
CGI	Common Gateway Interface
CPAN	Comprehensive Perl Archive Network
CSV	Comma Separated Value
DN	Distinguished Name
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
J2EE	Java Platform, Enterprise Edition
JSC	Jülich Supercomputing Centre
LDAP	Lightweight Directory Access Protocol
PDF	Portable Document Format
RCP	Rich Client Platform
SQL	Structured Query Language
SSL	Secure Sockets Layer
SVG	Scalable Vector Graphics
TLS	Transport Layer Security
URL	Uniform Resource Location
WWW	World Wide Web
XLS	Excel Spreadsheet
XML	Extensible Markup Language

Kapitel 1 Einleitung

Das *Jülich Supercomputing Centre (JSC)* ist weltweit eines der leistungsstärksten wissenschaftlichen Rechenzentren. Es ist im Forschungszentrum Jülich für die Planung, die Realisierung, den Betrieb und den Einsatz der Supercomputer, zentralen Serversysteme und Datennetze verantwortlich. Der Schwerpunkt liegt auf dem wissenschaftlichen Rechnen; die Supercomputer sind somit Instrument und Gegenstand der Forschung.

Die Rechnersysteme werden für vielfältige Anwendungen aus den Naturwissenschaften Physik, Chemie und Biologie z. B. für Computer-Simulationen genutzt. Dafür ist eine Vielzahl von Programmläufen erforderlich. Die Rechenzeit auf den Supercomputern muss daher auf zahlreiche Forschergruppen aufgeteilt werden. Um einen entsprechenden Anteil an der Nutzung zu garantieren, wird der betriebliche Leistungsprozess systematisch erfasst, überwacht und verdichtet. Dies geschieht durch das Accounting. Um Informationen über den Betrieb und die genutzten Ressourcen der Rechnersysteme zu erhalten, geben in diesem Zusammenhang Berichte darüber Aufschluss. Dazu soll ein Reporting-System erstellt werden, das einen schnellen und einfachen Überblick ermöglicht. Zudem soll es aber auch einen detailierten Einblick in die Daten erlauben. Daher ist es erforderlich, für die verschiedenen Nutzergruppen und die gewünschten Anforderungen bedarfsgerechte Zusammenstellungen zu erzeugen. Um dies zu gewährleisten, sollen die Nutzer selbst die Reports interaktiv erstellen und somit automatisch konfigurieren können. Die Ausgabe des Reports soll sowohl in tabellarischer als auch in grafischer Form möglich sein.

Ziel dieser Diplomarbeit ist der Entwurf eines Reporting-Systems für den Einsatz im JSC. Dabei soll die vorhandene Infrastruktur wie das Datenbank- und das Accounting-System berücksichtigt und darauf aufgebaut werden. Der Entwurf soll anhand einer konkreten Implementierung getestet werden.

In der vorliegenden Diplomarbeit werden im folgenden Kapitel die Grundlagen des Reporting beschrieben. Dazu wird der Begriff eingeordnet und abgrenzt. Weiterhin werden einige Grundgedanken zu Berichten und zur Technologie des Reporting dargelegt. Am Ende des Kapitels werden bekannte Reporting-Systeme vorgestellt.

Nach der allgemeinen Einführung wird in Kapitel 3 das Reporting im JSC beschrieben. Dabei werden zunächst die nachfolgend verwendeten Bezeichnungen eingeführt. Anschließend wird die derzeitige Vorgehensweise analysiert, wobei die existierenden Reports genauer untersucht werden. Aus der Problemanalyse werden dann die Anforderungen und Ziele des zu erstellenden Reporting-Systems gefolgert.

Im vierten Kapitel wird erläutert, wie durch die Realisierung einer Webanwendung ein einheitlicher und zentraler Zugang zu den Informationen geschaffen wird. Dazu werden verschiedene Möglichkeiten zur Architektur, zur Technologie, zum Ablauf und zur Sicherheit von Webanwendungen diskutiert.

Aufbauend auf der Analyse des derzeitigen Reporting und den Anforderungen und Zielen des neu zu erstellenden Systems wird in Kapitel 5 die Konzeptionierung beschrieben. In den einzelnen Abschnitten wird die Zugangs- und Zugriffskontrolle der Webanwendung erarbeitet und auf die Reporterstellung und deren Darstellung genauer eingegangen.

In Kapitel 6 wird die Realisierung des ausgearbeiteten Konzeptes dargestellt. Dazu werden die notwendigen Konfigurationen des Webservers beschrieben und die Auswahl der Programmiersprache begründet. Durch die Struktur und den Ablauf des Programmes wird die Implementierung des neu erstellten Reporting-Systems veranschaulicht.

Die Diplomarbeit wird mit einer Zusammenfassung in Kapitel 7 abgeschlossen. Dabei wird die Umsetzung der einzelnen Anforderungen und Ziele genauer betrachtet. Ein Ausblick soll Möglichkeiten von zukünftigen Weiterentwicklungen aufzeigen.

Kapitel 2 Reporting

Das zweite Kapitel dient zur allgemeinen Einführung in das Reporting. Dazu wird zuerst der Begriff des Berichtswesens definiert. Danach folgen Überlegungen zum Prozess der Berichterstellung, zur Gestaltung von Berichten und deren verschiedenen Einsatzbereiche. Der letzte Abschnitt gibt einen Überblick zu den bekanntesten Reporting-Systemen.

2.1. Einordnung des Berichtswesens

Das *Berichtswesen* (auch *Reporting*) ist ein wichtiges Instrument, um relevante Informationen zu betriebswirtschaftlichen Sachzusammenhängen zu erhalten. Die Entstehung und die Verwendung der Informationen fallen in einem Unternehmen meist (organisatorisch) auseinander. Als Bindeglied hat das Reporting nun die Aufgabe, notwendige Informationen und Daten für zukünftige Entscheidungen bereit zu stellen. Diese sollen für den Empfänger aufbereitet und auf das jeweilige Problem zugeschnitten erbracht werden. Die Informationen und Daten werden zum gewünschten Zeitpunkt und in geeigneter Form präsentiert.

In der Literatur (Jung, 2007) sind verschiedene Definitionen für das Berichtswesen zu finden:

- „Man versteht unter dem Berichtswesen alle Personen, Einrichtungen, Regelungen, Daten und Prozesse, mit denen Berichte erstellt und weitergegeben werden. Dabei stellen Berichte unter einer übergeordneten Zielsetzung, einem Unterrichtungszweck, zusammengefasste Informationen dar.“
- Das Berichtswesen ist „die Erstellung und Weiterleitung von funktionsübergreifenden Berichten im Sinne geordneter Zusammenstellungen von Nachrichten [...].“

Die Voraussetzung für die Entstehung eines Berichtes ist der Informationsbedarf. Dieser bezeichnet die Menge an Informationen, die der Berichtsempfänger für die Entscheidungsfindung benötigt. Sie dienen der Planung und Kontrolle. Im Berichtswesen werden die Berichte auch intern oder extern weitergegeben z.B. bei der Herausgabe des jährlichen Geschäftsberichtes.

2.2. Prozess der Berichterstellung

Der typische Prozess zur Erstellung von Berichten lässt sich in drei Schichten einteilen:

- Datenbereitstellung – Integration und Speicherung von Daten
- Analyse – methodische Auswertung
- Präsentation – Zugriff und Ausgabe

Die Basis bildet die Datenbereitstellung, in der das Datenmaterial zur Verfügung gestellt und zur weiteren Nutzung in den darauf folgenden Schichten verwendet wird. Dabei wird die persistente Speicherung, die meist durch den Einsatz von Datenbanken realisiert wird, unterschieden von der Aufbereitung der Daten z.B. zur Vereinheitlichung. In der Analyseschicht wird das Datenmaterial nach verschiedenen Kriterien untersucht und ausgewertet. Die Präsentation beinhaltet alle Funktionen für den Zugriff auf die Daten sowie verschiedene Anzeigetechniken

der Informationen. Die einzelnen Prozesse lassen sich jedoch im konkreten Anwendungsfall nicht immer genau trennen.

Das Reporting gliedert sich somit in die letzte der drei Schichten ein, da es auf den Zugriff und die Darstellung der Informationen ausgelegt ist. (Gluchowski, Gabriel, & Dittmar, 2008)

2.3. Gestaltung von Berichten

Berichte lassen sich nach unterschiedlichen Kriterien einordnen. Diese Kriterien können sowohl bei der Erfassung vorhandener Berichte als auch bei der Gestaltung neuer Berichte ermittelt werden. Zu den wesentlichen Merkmalen gehören neben dem Zweck, dem Inhalt und der Darstellung auch der zeitliche Aspekt und die betroffenen Instanzen. Je nach Einsatzbereich und Merkmal ergeben sich unterschiedliche Einzelkriterien, die im Folgenden beschrieben werden und in Abbildung 2-1 dargestellt sind.

Der *Berichtszweck* hat eine zentrale Bedeutung, da dadurch die anderen Merkmale mitbestimmt werden. Denn ist der Zweck eines Berichts nicht gegeben, stellt sich dessen Existenz in Frage. Als einen wichtigen Zweck kann die Dokumentation genannt werden. Die Berichte werden aber auch häufig zur Vorbereitung von Entscheidungen und zur Kontrolle herangezogen.

Der *Inhalt* eines Berichtes gibt den tatsächlichen Sachverhalt wieder. Dabei spielt die Nützlichkeit eine wesentliche Rolle. Die angebotenen Informationen sollen zweckmäßig detailiert sein und Vergleichsinformationen bereit stellen. Das Informationsspektrum sollte angemessen breit angelegt sein. Für die Interpretation der angebotenen Informationen sollte auch die Genauigkeit beachtet werden, da es sich möglicherweise um Abschätzungen handelt.

Die *Form* der Berichte kann entweder grafisch oder in Textform, einzeln oder gemischt dargestellt werden. Für Grafiken werden meist Achsendiagramme verwendet. Mit Tabellen und Listen können die Daten direkt ausgegeben werden. Die Berichtsinhalte sollten aus Gründen der Übersichtlichkeit und Verständlichkeit strukturiert angeordnet sein. Die verschiedenen Übertragungswege ergeben sich aus den gewählten Präsentationsmedien. Dabei wird zwischen der elektronischen Ausgabe und der auf Papier unterscheiden.

Bei dem *zeitlichen Aspekt* muss die Regelmäßigkeit beachtet werden. Wird ein Bericht in bestimmten Zeitintervallen angefertigt, handelt es sich um periodische Reports. Die aufbereiteten Informationen können sich auch auf unterschiedliche Betrachtungszeiträume beziehen z.B. auf einen Tag oder aber auf ein Jahr.

Als letztes Merkmal wird die *Instanz* angeführt. Von den Anforderungen der Empfänger hängt der Inhalt und Aufbau des Berichtes ab. Nur wenn exakte Informationen vorliegen, können diese an die verschiedenen Bedürfnisse angepasst werden. Für die Anfertigung der Berichte wird zwischen dem Verantwortlichen für den Inhalt und dem eigentlichen Ersteller unterscheiden. Die Berichte können manuell oder automatisch erzeugt werden.

Zwischen den verschiedenen Kriterien bestehen teilweise konkurrierende und komplementäre Beziehungen wie z.B. zwischen Detaillierung und Genauigkeit. Hierbei muss der Report nach dem Kriterium mit der höheren Priorität ausgerichtet werden. (Gluchowski, Gabriel, & Dittmar, 2008)

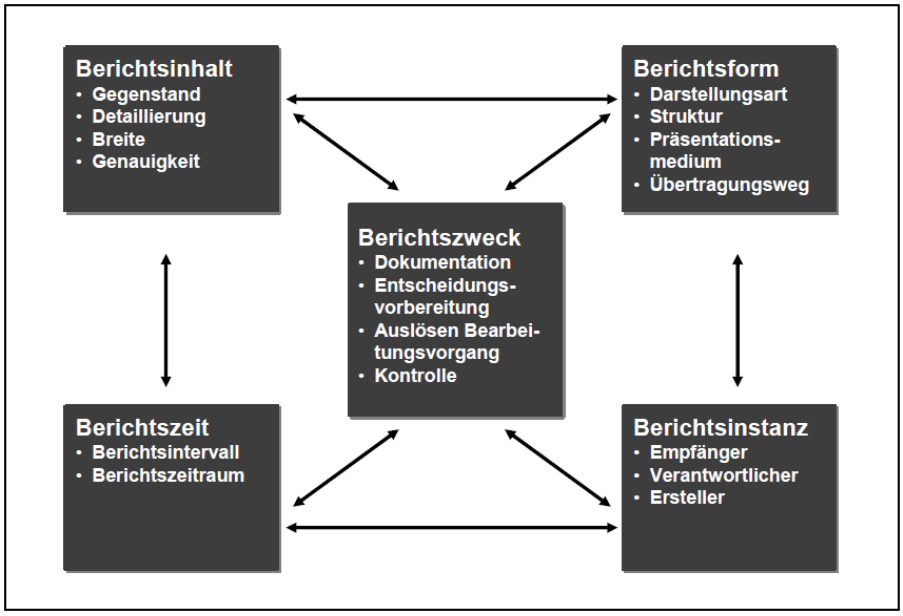

Abbildung 2-1 Merkmale zur Kennzeichnung und Gestaltung von Berichten[1]

2.4. Einsatzbereiche

Als Empfänger der Berichte kommen prinzipiell Fach- und Führungskräfte aller Bereiche eines Unternehmens und Hierarchieebenen sowie externe Empfänger in Betracht. Daher ist das Spektrum der Empfänger schlecht einzugrenzen. Es kann sich bei Berichten um Tages-, Wochen-, Monats- oder Jahresauswertungen aber auch Projektberichte handeln. Die Nutzergruppen haben also sehr unterschiedliche Anforderungen an die formale und inhaltliche Gestaltung.

Standardberichte werden periodisch nach festen Zeitintervallen erzeugt. Durch sie wird der allgemeine Bedarf an Informationen gedeckt. Häufig sind diese Berichte in fest vorgegebenen Formen und Inhalten identisch zu erstellen. Sie beziehen sich meist auf abgelaufene Perioden.

Durch den Eintritt bestimmter, vorab definierter Ereignisse werden sogenannte *Abweichungsberichte* erzeugt. Diese Informationen dienen zur Früherkennung bzw. Warnung. Die Berichte werden kurzfristig und unregelmäßig (aperiodisch) erstellt. Sie können damit verknüpfte Handlungen oder Vorgänge auslösen

Neben diesen beiden Berichten gibt es zusätzlich noch den *Bedarfsbericht*. Dieser wird nur in konkreten Situationen angefordert, z.B. wenn eine dringende Entscheidung ansteht. Aus diesem Grund sind der Inhalt und die Form vorab nicht definiert, sondern müssen im Einzelfall individuell festgelegt werden. Für die Erstellung dieser Berichte ist jedoch ein hoher Aufwand erforderlich. (Gluchowski, Gabriel, & Dittmar, 2008)

2.5. Reporting Technologien

Bei den unterschiedlichen Reporting-Lösungen lässt sich die Sichtweise des Entwicklers von der der eigentlichen Benutzer abgrenzen. Prinzipiell beinhaltet jeder Bericht eine abstrakte

[1] Aus (Gluchowski, Gabriel, & Dittmar, 2008) S.208

Schablone mit unterschiedlichen Gestaltungselementen und ein konkretes Berichtsergebnis. Wie eine Report-Schablone angelegt wird, hängt von der Report-Generierung ab. Im einfachsten Fall werden mit Hilfe einer höheren Programmiersprache Sprachkonstrukte für formatierte und grafische Darstellung erzeugt. Aus heutiger Sicht wird aber eine Software zum Zusammenstellen der Berichte ohne Programmierung bevorzugt. Somit können diese auch von den Endanwendern und nicht nur von Softwareentwicklern erstellt werden. Durch das Abspeichern der Reporting-Schablonen lassen sich diese beliebig oft wiederverwenden z.B. bei Standardberichten. Konkrete Auswertungsergebnisse sollen jedoch nur bei zeitaufwendigen Berechnungen oder zu Dokumentationszwecken gesichert werden.

Bei einer technischen Lösung des Berichtssystems werden die Berichte in elektronischer Form erzeugt und präsentiert. Wesentlich für elektronische Standardberichte ist, dass die Berichte weitgehend automatisch generiert werden. Sie verknüpfen die Informationen und bereiten diese empfängergerecht auf. Als *Berichtssystem* oder auch *Reporting-System* werden „diejenigen Software-Werkzeuge bezeichnet, die über die reine Abfragefunktionalität hinaus noch die Möglichkeit zur inhaltlichen und optischen Anreicherung bieten. Die Einsatzbereiche dieser Werkzeuge reichen von der Definition und Anzeige von Standardberichten über die Erstellung von Ad-Hoc-Auswertungen bis zur freien Navigation im Datenbestand“ (Gluchowski, Gabriel, & Dittmar, 2008). Es gibt zwei verschiedene Typen von Reporting-Systemen:

- Query-basiert: Innerhalb vordefinierter Datensichten kann der Benutzer die Daten selektieren und präsentieren.
- Tabellen-basiert: Hier werden die Datensichten selbst zusammengestellt. Dabei ist zu beachten, dass der Benutzer wissen muss, welche Daten in welcher Tabelle stehen. (Wikipedia, Die freie Enzyklopädie, 2007)

2.6. Reporting Tools

Der folgende Abschnitt nimmt zunächst eine Einordnung und Abgrenzung des Begriffs Business Intelligence vor. Anschließend werden die bekanntesten Reporting-Tools in ihrer Funktionsweise und Architektur einzeln erläutert. Ein Fazit zum Einsatz dieser Tools schließt das Unterkapitel ab.

2.6.1. Business Intelligence

Im Bereich des Reporting ist der Begriff der *Business Intelligence (BI)* immer wieder zu finden, er entstand am Anfang der 90er Jahre. Intelligence bedeutet in diesem Zusammenhang jedoch nicht Intelligenz, sondern wird im Sinne von Einsicht oder Verständnis interpretiert. Nach (Gluchowski, Gabriel, & Dittmar, 2008) „handelt es sich hierbei um Komponenten zur Extraktion, Bereinigung, Transformation, Integration, Speicherung und entscheidungsorientierte Aufbereitung relevanter Informationen sowie um die Bausteine zur Präsentation und Analyse dieser Inhalte.“ Mittels geeigneter Software lassen sich also aus riesigen, oft unstrukturierten Datenmengen Informationen gewinnen aber auch Prognosen berechnen und ableiten. BI-Systeme bieten alles in einem Komplettpaket an (Kleijn, 2006). Im Moment gibt es drei im Open-Source-Markt führende Tools: *Pentaho BI Suite*, *BIRT* (Business Intelligence and Reporting Tools) und *JasperReports*.

In den folgenden Abschnitten werden nicht die gesamten Business Intelligence Systeme erläutert, sondern nur die einzelnen Reporting-Lösungen dieser Systeme betrachtet. Die Reportentwicklung ist bei allen drei Systemen ähnlich, sie unterteilt sich immer in zwei Stufen: die *Design-Time* und die *Run-Time*. Durch diese Aufteilung wird die Run-Time auf einem Server oder in einer externen Anwendung platziert und so von der eigentlichen Reportentwicklung getrennt.

2.6.2. Pentaho-Reporting

Pentaho-Reporting ist ein Modul der kompletten Pentaho-BI-Suite, die aus vielen einzelnen Open Source Projekten entstanden ist. Es nutzt für das Reporting die Open Source Java Bibliothek JFreeReport.

Bei Pentaho besteht die Möglichkeit einen Bericht durch den Report-Designer oder durch den Report-Design-Wizard zu entwerfen. Der Bericht wird dabei als eine einfache XML-Template-Datei gespeichert und muss für die Reporting-Engine noch exportiert werden. Dabei werden zwei Dateien erstellt: eine Action-Datei, in der die Aktionen zur Berichterzeugung definiert sind, und eine Extended-Report-Definition-Datei, in der das komplette Report-Objekt im XML-Format beschrieben wird. In der Run-Time wird die Action-Datei interpretiert und die entsprechenden Aktionen ausgeführt. Wird nicht die gesamte Plattform verwendet, sondern nur die Bibliothek, ist die Action-Datei unnötig. In diesem Fall muss dem Report-Objekt aber die Verbindungsdaten zur Datenbank mitgegeben werden.

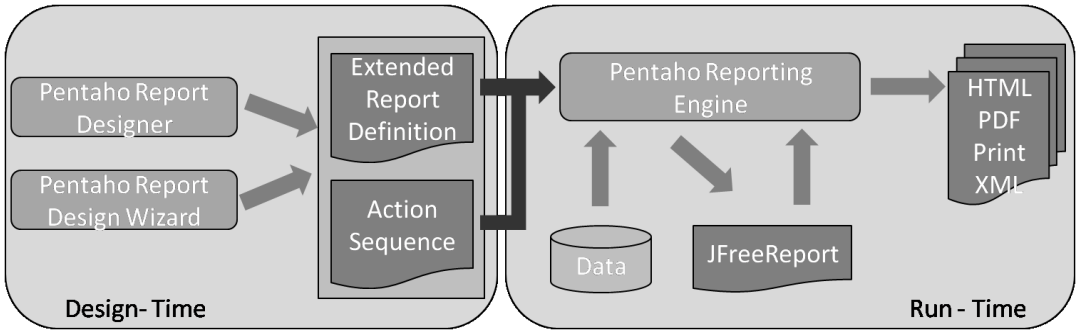

Abbildung 2-2 Report-Entwicklung von Pentaho-Reporting

Pentaho verwendet eine J2EE-Architektur, die prozessorientiert aufgebaut ist. Es werden hierbei Business Intelligence Prozesse ausgeführt, die durch Action-Sequences in XML beschrieben sind. Ein einfacher Report lässt sich durch drei Aktionen definieren: Abfrage der Parameter, Abfrage der Daten aus der Datenquelle und die Aufbereitung und Ausgabe von einem Bericht. Durch die modulare Architektur können die Module auch ausgetauscht oder Alternativen verwendet werden. Somit wird eine hohe Flexibilität erreicht, jedoch zum Nachteil der Übersichtlichkeit. [(Doumack, 2008), (Held & Klose, 2007), (Pentaho Open Source Business Intelligence, 2009)]

2.6.3. JasperReport

JasperReport ist die Reportlösung der Firma JasperSoft und wird als eine Open-Source-Java-Bibliothek mit einem Application Programming Interface (API) angeboten. Eine API ist eine Schnittstelle, die es ermöglicht von einem Programm aus auf bestimmte Funktionen eines anderen Softwaresystems zu zugreifen.

Bei JasperReport kann der Designer IReport oder ein Eigener zur Berichterstellung benutzt werden. In der Design-Time wird der Bericht in einer Datei als XML-Report-Template gespeichert. Dieser kann gleichzeitig oder erst in der Run-Time kompiliert werden. Das Kompilieren kann aber auch von Hand oder durch andere Werkzeuge wie ANT (Another Neat Tool) realisiert werden. In der Run-Time holt die Jasper-Reporting-Engine die benötigten Daten aus den Datenquellen und bereitet diese auf. Als Ergebnis davon wird das Report-Dokument gespeichert, das zur Betrachtung, zum Drucken oder zum Exportieren in verschiedene Formate geeignet ist. Dies kann von der Engine aber auch selbst durchgeführt werden.

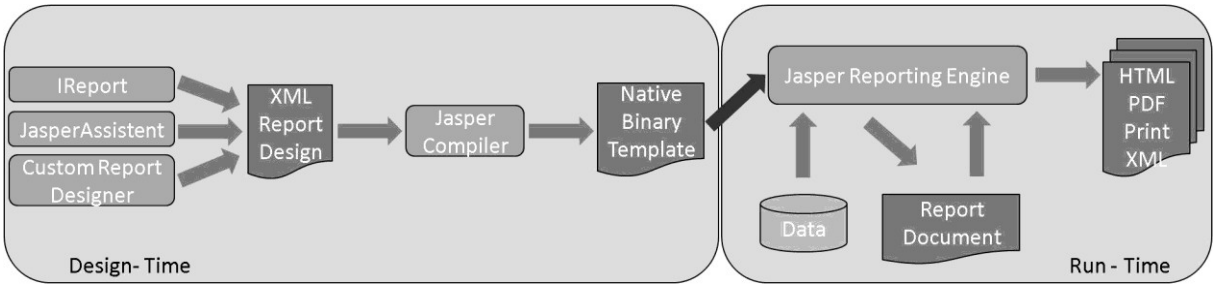

Abbildung 2-3 Reportentwicklung von JasperReports

Ein Vorteil der Architektur der JasperReports-Bibliothek ist, dass sie modular aufgebaut ist und sich dadurch in jede Java-Anwendung integrieren lässt. Für einige Ausgabe-Formate werden Fremdbibliotheken aus anderen Open-Source Projekten benutzt. Vorteilhaft ist außerdem, dass Jasper-Report eine API zur Verfügung stellt, um bestimmte Funktionalitäten zu erweitern oder zu verändern wie z.B. den Datenbankzugriff. Mit Jasper-Design wird das Rohmaterial des Berichtes in das XML-Report-Template gebracht. Der Compile-Manager wird von Jasper-Report aufgerufen. Er stellt alle Kompilierungsfunktionalitäten zur Verfügung, um aus dem Design einen Report zu erzeugen. Der Fill-Manager ist dann für die Datenaufbereitung zuständig. Als Ergebnis entsteht eine Instanz von Jasper-Print, die das proprietäre Format zur Speicherung von vollfunktionsfähigen Dokumenten darstellt. Mit dem Export-Manager werden Methoden zum Transformieren in verschiedene Formate angeboten. Der Print-Manager bietet Funktionen zum Anzeigen und Drucken. [(Doumack, 2008), (JasperSoft, 2009)]

2.6.4. BIRT

BIRT ist ein auf Eclipse basierendes Open Source Projekt, das eine Komplettlösung zum Erstellen von Berichten anbietet.

Bei BIRT wird in der Design-Time der Bericht über den vorhandenen oder einem eigenen Designer entworfen und an die Datenquelle angebunden. Die Designer verwenden dazu die API von der Report Design Engine. Die Informationen werden in der Report-Design-Datei gespeichert, die auf der Syntax von ROM (Report Object Model) in XML basiert. In der Run-Time wird dann der Bericht durch die Design-Datei erstellt. Dieser Prozess läuft in zwei Phasen ab: Die erste Phase ist die Generierung, in der die Datenquellen abgefragt, die Daten aufbereitet und in dem Report-Document gespeichert werden. In der zweiten Phase, der Präsentation, wird dieses dann gerendert und das gewünschte Format z.B. PDF erzeugt.

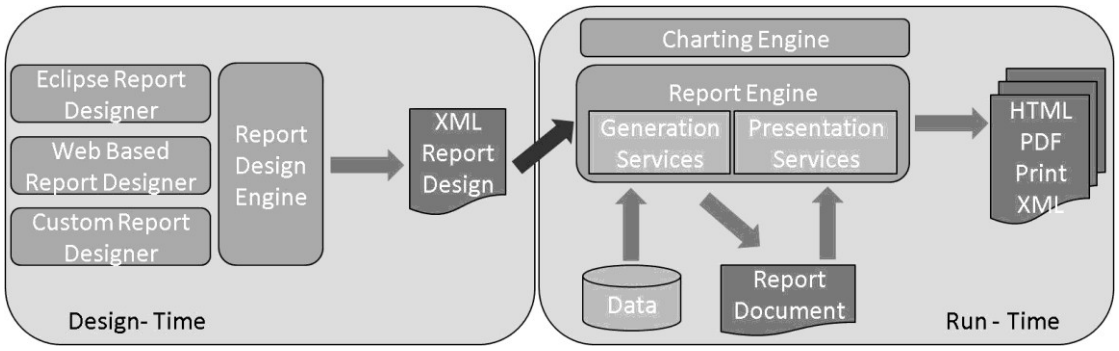

Abbildung 2-4 Reportentwicklung von BIRT

Die Architektur ist sehr eng in die Architektur von Eclipse integriert. Es folgt dem Eclipse-Konzept und besteht somit aus Plug-Ins. BIRT bietet drei Hauptapplikationen: BIRT Report Designer, BIRT RCP Report Designer und BIRT Report Viewer. Der Report-Designer ist ein grafisches Werkzeug um Berichte zu entwerfen. Unter der Verwendung der Report-Design-Engine erzeugt der Designer die Report-Design-Datei. Der RCP Report-Designer ist eine eigenständige Rich-Client-Platform-Anwendung, die ein vereinfachtes Design-Interface besitzt. Der Report Viewer ist ein Java-Web-Servlet, der aus dem Report-Design einen Bericht generiert und darstellt.

BIRT verwendet mehrere Engines, die eine API anbieten, um verschiedene Funktionalitäten bereit zu stellen:

- Die Chart-Engine enthält APIs zur Generierung von Diagrammen und verbindet diese mit den Daten.
- Die Report-Design-Engine besitzt APIs zur Validierung und Generierung von Report-Design-Dateien.
- Die Data-Engine beinhaltet APIs zum Aufbereiten und Abrufen der Daten. Sie besteht aus der Data-Access-Komponente, die die Daten liefert, und der Data-Transform-Komponente, die die Daten sortiert, gruppiert, aggregiert und filtert.
- Die Report-Engine teilt sich in die Generation- und die Presentation-Engine. Die Generation-Engine enthält die APIs zum Auslesen und Interpretieren der Report-Design-Dateien und erzeugt das Report-Document. Die Presentation-Engine verwendet das Document und erstellt daraus einen Bericht im gewünschten Format. [(Doumack, 2008), (Eclipse Foundation, 2009)]

2.6.5. Fazit

Die Reporting-Tools bieten eine breite Funktionsvielfalt. Zu den Standardfunktionen im Reporting-Bereich zählen die Gruppierung von Daten, die Anzeige von Header und Footer sowie die Generierung von Berichten im HTML- und PDF-Format. Daneben können die Reports mit Teilreports verschachtelt, die Berichte parametrisiert und mehrerer Datenquellen innerhalb eines Berichtes verwendet werden. Sie lassen also kaum Wünsche offen. Daraus ergeben sich jedoch auch einige Probleme:

Tabellen- und Feldbezeichnungen präsentieren sich als kryptische Abkürzungen, die nicht immer auf den Inhalt schließen lassen. Ausführliche Dokumentationen der Datenbankstruktur sind

kaum benutzergeeignet. Der Benutzer muss eine genaue Kenntnis der Tabellenstruktur der Datenbank besitzen, um SQL-Abfragen zu entwickeln und um mehrere Tabellen miteinander verknüpfen zu können. Der Anwender eines Reporting Tools kann durch zu viele Einstellungsmöglichkeiten beim Design der Diagramme leicht überfordert werden. Der Entwurf der Berichte ist also sehr umfangreich. Der volle Funktionsumfang zur Report-Erstellung ist bei Web-Applikationen bisher nicht gegeben, sondern nur bei den Desktop-Anwendungen, die vorher installiert werden müssen. Damit steigen auch Speicherverbrauch und Auslastung des jeweiligen Arbeitsplatzrechners. Die marktüblichen Reporting- Tools sind zudem so komplex aufgebaut, dass sie ohne Handbuch oder ein Tutorial nur eingeschränkt nutzbar sind. Unter Umständen muss professioneller Support in Anspruch genommen werden. Andernfalls ist mit einer umfangreichen Einarbeitungszeit zu rechnen. BI-Systeme sind erst erfolgreich, wenn das Wissen aus verschiedenen Bereichen mit flexiblen Analysen zur täglichen Nutzung kombiniert wird. Dazu sollte zum Aufbau einer BI-Anwendung eine Arbeitsteilung zwischen den Fachbereichen und der IT erfolgen. Bei den derzeitigen BI-Systemen wird jedoch nur ein geringer Stellenwert darauf gelegt. Somit sind entweder Insellösungen oder unzureichende Informationsqualität die Folge. [(Gluchowski, Gabriel, & Dittmar, 2008), (Doumack, 2008), (Kleijn, 2006), (Finger, 2006)]

Kapitel 3 Reporting im JSC

In Kapitel 2 wurde der allgemeine Begriff des Reporting eingeführt und die bekanntesten Reporting-Tools erläutert. In diesem Kapitel folgt die nähere Beschreibung der Berichterstellung im JSC. Dabei werden die in dieser Arbeit verwendeten Bezeichnungen in ihrer Bedeutung erklärt. Anschließend wird die derzeitige Vorgehensweise analysiert, die daraus resultierenden Probleme aufgezeigt und die Anforderungen und Ziele des zu erstellenden Reporting-Werkzeuges diskutiert.

3.1. Begriffsbestimmungen

Im Jülich Supercomputing Centre wird die *Rechenkapazität* für Wissenschaftler im Forschungszentrum, an Universitäten und Forschungseinrichtungen in Deutschland und zum Teil in Europa sowie der Industrie über das John von Neumann Institut für Computing (NIC) bereitgestellt. Um auf einem Rechnersystem arbeiten zu können, muss innerhalb eines *Projektes* die Rechenzeit vorher beantragt werden. Sie kann grundsätzlich von jedem fachlich entsprechend ausgewiesenen Wissenschaftler in Deutschland sowie Partnern von EU-Projekten beantragt werden. Die Vergabe von Rechnerressourcen wird anhand unabhängiger Gutachten (Peer-Review-Verfahren) und in Anlehnung an die Kriterien und Verfahrensweisen der Deutschen Forschergemeinschaft vorgenommen. Dabei ist die wissenschaftliche Exzellenz das wichtigste Kriterium.

Dem Projekt wird bei der Bewilligung ein *Projektbetreuer* zugeordnet. Er ist Mitarbeiter im JSC und gleichzeitig fachlicher Ansprechpartner für ein Projekt. Der Projektbetreuer nimmt die Probleme der Nutzer entgegen und berät sie.

Die bewilligte Rechenzeit für ein Projekt steht dem Wissenschaftler in Form eines *Kontingentes* zur Verfügung. Diese Zuteilungen sind entweder auf die gesamte Rechenzeitperiode oder pro Monat festgelegt. Für die *Teilnehmer eines Projektes* werden auf den Rechnersystemen *Accounts* eingerichtet. Somit steht jedem Benutzer auf dem Rechensystem eine Benutzer-ID zur Verfügung. Neben den Kontaktdaten eines Benutzers sind ebenso seine verschiedenen Accounts in der Datenbank gespeichert.

Die Rechenzeit wird auf viele verschiedene Forschergruppen aufgeteilt. Diese nutzen die Rechnersysteme z. B. für Computer-Simulationen, wofür eine Vielzahl von *Rechenjobs* erforderlich ist. Ein Job stellt dabei einen Prozess dar, der dem Nutzer zugeordnet ist.

Als *Supercomputer* sind das IBM Power6 Cluster *JUMP* (Juelich Multi-Processor) und die IBM Blue Gene P *JUGENE* (Juelicher Blue Gene P) im Einsatz. Zudem gibt es noch verschiedene Server- und Cluster-Systeme. In der Abbildung 3-1 sind die Rechnerressourcen des JSC dargestellt.

Die *Kontingentierung* dient zur Verteilung der vorhandenen Ressourcen gemäß der Vorgaben des Managements. Dadurch wird eine Zuteilung auf verschiedene Benutzergruppen bzw. Projekte ermöglicht, die einen entsprechenden Anteil an der Nutzung garantiert.

Mit Hilfe des *Accounting* wird der betriebliche Leistungsprozess systematisch erfasst, überwacht und verdichtet. Im JSC wird dadurch der Nachweis über die erbrachte Rechenleistung

geführt. Insbesondere werden von dem *Accounting-System* alle notwendigen Daten zu einem Job basierend auf den Log-Dateien in einer Oracle-Datenbank gespeichert.

Das *Reporting* im JSC soll nun Aufschluss über den Betrieb und die genutzten Ressourcen der Rechnersysteme geben. Somit ist die Auslastung der Rechner besser planbar, da der Verbrauch der Rechenzeit-Kontingente erfasst wird. Dafür werden Berichte in den verschiedensten Formen und Darstellungen erstellt. Das Reporting ist also eine wichtige Komponente im produktiven Betrieb des JSC.

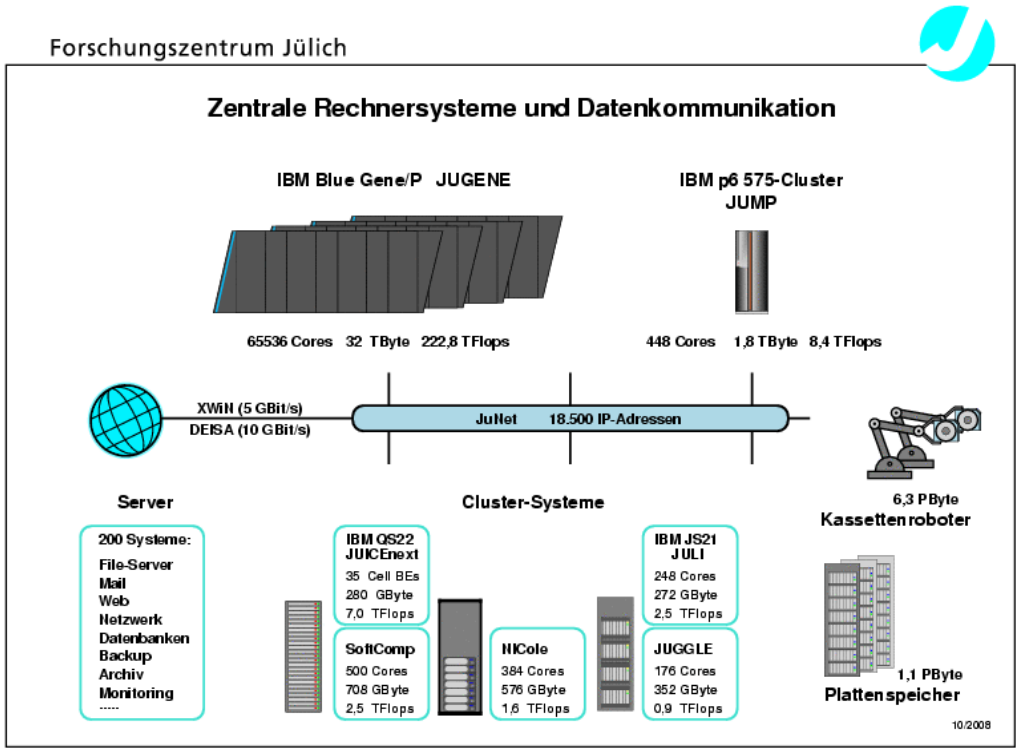

Abbildung 3-1 Zentrale Rechnersysteme des JSC

3.2. Reports

Im folgenden Abschnitt werden die verschiedenen Reports beschrieben und analysiert. Dabei werden alle Berichte aufgeführt, die zurzeit erstellt werden. Sie sind nach unterschiedlichen Einsatzbereichen gegliedert.

3.2.1. Monats- und Jahresauswertungen

Vom Administrator der Accounting-Software werden für verschiedene Benutzergruppen und auch auf Anfragen einzelner Benutzer Berichte erstellt. Insbesondere handelt es sich dabei um monatliche und jährliche Auswertungen für das Management und die Systemadministratoren. Die monatlichen Reports werden immer am darauf folgenden Monatsanfang angefertigt. Sie umfassen u. a.:

- Verbrauchsstatistiken der Benutzer in Verrechnungseinheiten und Kosten für die Verwaltung
- Übersichten über die Verfügbarkeit, Auslastung, Anzahl der Jobs der Maschinen für das Management

- Statistiken über die Auslastung der Rechner, Wartungs- und Ausfallzeiten für die Systemadministratoren.

Am Jahresanfang werden die monatlichen Auswertungen zusammengefasst und in Jahresberichten veröffentlicht. Sowohl die monatlichen als auch die jährlichen Reports dienen der Dokumentation und gleichzeitig auch der Kontrolle. Die Daten in den Berichten sind wegen dem relativ langen Berichtszeitraum monatlich bzw. jährlich zusammengefasst. Dadurch gehen natürlich die Genauigkeit und die Detaillierung der Daten verloren. Dies ist aber für die monatlichen Auswertungen ausreichend. Weiterhin werden für einige Daten nur die Prozentwerte angegeben, so dass der tatsächliche Wert nicht angezeigt wird, es dient der Überschaubarkeit.

Die Reports stehen auf den internen JSC-Webseiten zur Verfügung oder werden direkt an das Management weitergeleitet. Auf die internen Webseiten können nur befugte Personen wie z.B. die Systemadministratoren zugreifen. Der Zugriff wird durch einen Login mittels Emailadresse und Passwort geregelt. Die Zugangsdaten sind auf dem Webserver in einer Datei gespeichert, so dass diese für das Authentisieren genutzt werden kann. Die Weiterleitung an das Management erfolgt meist per Email an den jeweilig Empfangsberechtigten. Die Email wird durch den Administrator der Accounting-Software erstellt und versendet.

Bei den Monats- und Jahresauswertungen handelt es sich um periodische Standardreports. Diese werden immer nach dem gleichen Schema erzeugt. Anhand des folgenden Beispiels *Rechenzeitverteilung der Projekte* kann die Erstellung dieser Berichte nachvollzogen werden:

1. Die Daten für die Diagramme werden durch SQL-Abfragen der Datenbank generiert. Dabei werden die Daten entsprechend gruppiert und zusammengefasst. Die notwendigen Abfragen für das genannte Beispiel sind in der Abbildung 3-2 zu sehen. Die erste Abfrage ermittelt die gesamte Rechnerauslastung ab dem 1. Dezember 2006 und die Zweite die Auslastung der einzelnen Projektgruppen.

```
SQL>  select sum(walltime)/3600, sum(wallcpu)/3600 from sc_jobs_m where
sepjob in ('G','N') and enddate>='01-DEC-06';

SUM(WALLTIME)/3600 SUM(WALLCPU)/3600
------------------ -----------------
           3644474           4755896

SQL>  select groupid,sum(walltime)/3600,sum(wallcpu)/3600 from sc_jobs_m
where sepjob in ('G','N') and enddate>='01-DEC-06' group by groupid
order by 3;

GROUPID  SUM(WALLTIME)/3600 SUM(WALLCPU)/3600
-------- ------------------ -----------------
unknown            63.52833          63.52833
group7            125.8697          125.8697
group8            349.0414         2800.921
group3          130367.5          130367.5
group1          114962.3          361553.7
group6          192595.2          467088.3
group5          319867.7          544149.6
group4         1230024           1270192
group2         1656119           1979555
```

Abbildung 3-2 SQL-Abfragen zur Erstellung des Berichtes

2. Die Daten müssen für die grafische Darstellung eventuell noch aufbereitet bzw. umgerechnet werden. Diese werden dann in einer Daten-Datei gespeichert. In der Abbildung 3-3 ist in der Y-Spalte der prozentuale Anteil der jeweiligen Projektgruppe am Gesamtverbrauch zu sehen.

```
DATA_LABELS        X          Y
group1             1       7.60
group2             2      41.62
group3             3       2.74
group4             4      26.71
group5             5      11.44
group6             6       9.82
other              7       0.06
all                8     100
```

Abbildung 3-3 Daten-Datei des Berichtes

3. Für das Erzeugen der Diagramme wird das Graphikprogramm Gsharp von AVS (Advanced Visual Systems Inc.) genutzt. Gsharp ist ein Programm zur wissenschaftlichen Datenvisualisierung. Es dient zur Erstellung von hochwertigen Präsentationsgraphiken (Graphen und Diagramme) im 2D/3D-Bereich. Über eine Oberfläche werden die Grafiken interaktiv oder durch zuvor erzeugte und editierbare Gsharp Skripte erstellt. Da es sich bei den Grafiken um Standardreports handelt, werden immer dieselben Gsharp-Skripte aufgerufen. Um die Grafik zu erzeugen, muss also nur die Daten-Datei von dem Gsharp-Skript eingelesen werden. Die Abbildung 3-4 zeigt das von Gsharp erstellte Diagramm des Beispiels.

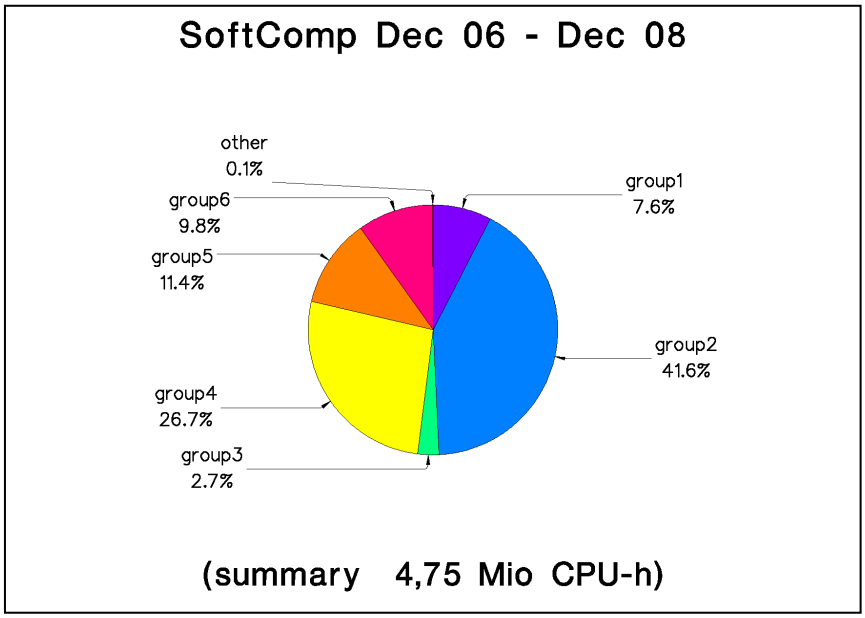

Abbildung 3-4 Grafik des Berichtes „*Rechenzeitverteilung der Projekte*“

3.2.2. Top-User-Listen

Die Top-User-Listen werden täglich per Email verschickt, sie können aber auch jederzeit erzeugt werden. Sie werden sowohl für das Management also auch das *Parateam* erstellt, das die Nutzung der Supercomputer- und Serversysteme überwacht. Die Listen werden somit zur täglichen Kontrolle und Dokumentation genutzt, insbesondere zur Anzeige der Aktivität auf den Rechnersystemen.

Erzeugt werden sie durch ein komplexes Perl-Programm. In diesem wird ein Skript aufgerufen, dem als Kommando-Befehl *qquis* mit unterschiedlichen Optionen zur Verfügung steht. Auch hier werden wieder die gewünschten Daten über Datenbank-Abfragen ermittelt. Der Befehl ruft standardmäßig die Personendaten, die Accounting-Angaben, das Rechnersystem, den Abrechungszeitraum und alle Informationen zum jeweiligen Kontingent ab. Zurückgegeben wird dann eine durch Komma getrennte Liste. Mit diesen Daten werden zusätzlich weitere Berechnungen durchgeführt z.B. der lineare Verlauf des Verbrauches. Die einzelnen Kontingent- und Rechenzeit-Verbräuche werden summiert bzw. zusammengefasst und daraus die abgelaufene und verbleibende Rechenzeit berechnet. Somit wird eine Prognose zur laufenden Kontingentperiode erstellt. Am Ende werden alle Informationen und berechneten Daten nach den gewünschten Anforderungen sortiert und formatiert. Die Daten werden entweder als ASCII-Datei ausgegeben oder in eine PDF-Datei geschrieben. Diese täglich erzeugten Listen werden separat gespeichert. Das Programm erstellt mehrere Listen, hier werden nur die wichtigsten genannt und kurz erläutert:

- Eine der wichtigsten Top-User-Listen beinhaltet die verbrauchten Projektkontingente, die absteigend sortiert sind. Es werden dazu der Projektname, der Kontingentpool, der Top-User des Projektes ausgegeben. Weiterhin werden die verbrauchten Kontingenteinheiten und die genutzte Rechenzeit im Vergleich zum Gesamtkontingent bzw. zur Gesamtrechenzeit aufgelistet. Am Ende wird noch der Trend der weiteren Nutzung berechnet, falls der Verbrauch gleichbleibend ist.
- Andere Listen beziehen sich auf die Top-User, die jeweils am letzten Tag, in der letzten Woche oder im aktuellen Monat das meiste Kontingent verbraucht haben. Dabei wird der jeweilige Benutzername und sein Account, die verbrauchte Rechenzeit in Bezug zur gesamt verfügbaren Zeit, das genutzte Kontingent und die jeweiligen Jobgrößen aufgelistet.
- Eine weitere Liste informiert über alle Benutzer und deren Kontingent- und Rechenzeit-Verbrauch. Dabei werden sogenannte Quantile eingefügt, so dass deutlich wird, welche Nutzer am aktivsten auf dem Rechnersystem arbeiten.

In einer Datei sind dann alle Informationen zu einem Rechnersystem zusammengefasst und in Tabellen dargestellt. Die Datei beinhaltet also eine Vielzahl von verschiedenen Tabellen mit unterschiedlichen Informationen. So muss sich der Benutzer erst die Tabelle mit den gewünschten Informationen suchen. Der Inhalt der Tabellen ist teilweise sehr komplex und somit wird die Übersichtlichkeit beeinträchtigt.

Da diese Listen per Email an einen autorisierten Empfängerkreis versendet werden, ist keine Authentisierung mehr notwendig. Ein Auszug der Listen soll im Anhang unter dem Abschnitt Top-User-Listen einen Einblick geben. Dabei werden die drei obengenannten Listen aufgeführt.

3.2.3. Rechenzeit-Kontingent per Email

Die Nutzer eines Projektes und die Projektleiter werden per Emails über ihr Rechenzeit-Kontingent informiert. Diese Emails werden regelmäßig am Monatsanfang durch das Kontingentierungs-System versendet. Sie beinhalten den aktuellen Kontingentstand, das verbleibende Kontingent sowie eine Detailübersicht über den Verbrauch auf den verschiedenen

Rechnersystemen. Weiterhin werden Mitteilungen an das Projekt verschickt, wenn sich Kontingente ändern oder verbraucht sind. Somit ergeben sich zwei unterschiedliche Verwendungsmöglichkeiten: Einerseits wird das Rechenzeit-Kontingent zur Dokumentation an die Empfänger geschickt. Andererseits wird durch die Tatsache, dass das Projektkontingent verbraucht bzw. überschritten wird, der Benutzer durch diese Emails gewarnt. Die Reports werden also genau in diesem Moment erstellt. Dabei beinhaltet eine Mitteilung an das Projekt den Kontingentpool, den Abrechnungszeitraum, die bewilligten Kontingenteinheiten, den aktuellen Gesamtverbrauch und das verbleibende Kontingent. Der Prozess der Berichterstellung und das Versenden der Emails werden durch das Kontingentierungs-System automatisch durchgeführt. Die Daten werden wiederum durch verschiedene Datenbank-Abfragen generiert und zusammengefasst, so dass die Daten an die Bedürfnisse der Benutzer bereits angepasst sind. Der monatliche Kontingentstand lässt sich somit auch immer vergleichen. Bei der Rechenzeit-Kontingentierung gibt es nur eine Textausgabe und keine grafische Darstellung. Durch das Versenden der Emails entfällt die separate Authentisierung. In den Abbildung 3-5 und Abbildung 3-6 sind Auszüge solcher Emails zu sehen. Zuerst wird die monatliche Email für die Projektleiter dargestellt und danach die Mitteilung an das Projekt, wenn ihr Kontingent verbraucht ist.

```
Sehr geehrte/r Projektleiter/in des Projekts XXX,

Diese Mail dient als Information ueber den derzeitigen Kontingentstand Ihres
Projekts auf den Rechnersystemen des JSC.

Abrechnungszeitraum : 01.07.2008 bis 30.06.2009


Kontingentpool      : xxxxx
==================================

Kontingentstand auf dem System JUMP
----------------------------------

Verrechnungsart : Festkontingent
Bewilligte Kontingenteinheiten :        5335 KE
Gesamtverbrauch                :        653.6 KE
Verfuegbares Kontingent in %   :         87.7

Detailuebersicht ueber den Verbrauch auf JUMP:

UserId   Benutzername                         KE gesamt
-----------------------------------------------------
xxxxx    xxxxxxxxxx, xxxx                         0.0
xxxxx    xxxxx, xxxxxx                            0.0
xxxxxxx  xxxxxxx, xxxxx                           0.0
xxxxxxx  xxxxxx, xxxxxx                           0.0
xxxxxx   xxxxxx, xxxxxxxxxx                       0.0
xxxxxxx  xxxxxxx, xxxx                          653.6
xxxxxx   xxxxxxxxxx, xxxxxxxx                     0.0


Mit freundlichen Gruessen,
Ihr DISPATCH
```

Abbildung 3-5 Email zum Kontingentstand eines Projektes

```
Mitteilung an das Projekt XXX,

das Kontingent Ihres Projektes auf dem IBM-Supercomputer Jump
ist verbraucht. Die aktuellen Daten sind:

Kontingentpool  : xxxxxx
Verrechnungsart : Festkontingent

Abrechnungszeitraum            : 01.07.2008 bis 30.06.2009

Bewilligte Kontingenteinheiten :          30 KE (1 CPU-h=.3 KE)

Gesamtverbrauch                :         81.1 KE

Verfuegbares Kontingent in %   : -170.5

[...]

Mit freundlichen Gruessen,
Ihr DISPATCH
```

Abbildung 3-6 Email zum Kontingentverbrauch eines Projektes

3.2.4. Kontingentinformationen im Web

Die Projektbetreuer und die Projektleiter können eine Kontingentliste bzw. den Kontingentstand ihrer Projekte auch über das Web einsehen. Somit wird den Projektleitern eine weitere Möglichkeit gegeben, Projektinformationen zu erhalten. Dadurch, dass die Angaben im Internet zur Verfügung stehen, können sie jedoch zu jeder Zeit darauf zugreifen. Dies muss allerdings aus Eigeninitiative erfolgen, der Benutzer hat also gewissermaßen eine Hol-Schuld. Diese Berichte dienen zur Information und Kontrolle. Die Reports werden automatisch durch eine Webanwendung erstellt, die in TCL programmiert wurde.

Für den Login müssen sich die Projektleiter mit ihrem Projektnamen und dem Projekt-Passwort anmelden. Das Passwort wird dem Projektleiter beim Beantragen eines Projektes mitgeteilt und gleichzeitig auch in der Datenbank gespeichert. Die Autorisierung des Projektleiters erfolgt hier über eine Datenbank-Abfrage. Es besteht aber auch die Möglichkeit, sich per Email zu verifizieren. Dazu wird an den zu einem Projekt eingetragenen Projektleiter eine Email versendet. Diese enthält dann die URL der Webanwendung mit den benutzerspezifischen Parametern.

Die Projektleiter und Projektbetreuer, die im Forschungszentrum Jülich eine Email-Adresse haben, können sich auch mit Hilfe ihres Email-Passworts anmelden. Das Login-Fenster wird in Abbildung 3-7 dargestellt.

Die Projektbetreuer können außerdem über den Benutzerverwaltungs-Zugang an die Projekt-informationen gelangen. Dafür brauchen sie nur den gewünschten Projektnamen eingeben. Der Dispatch-Zugang steht nur auf den internen JSC-Webseiten zur Verfügung. Der Zugriff wird durch einen Login mittels Emailadresse und Passwort geregelt. Die Zugangsdaten sind auf dem Webserver in einer Datei gespeichert, so dass diese für das Authentisieren genutzt werden kann.

Abbildung 3-7 Login für die Kontingentinformationen im Web

Danach werden die Kontingentinformationen zu einem Projekt wieder über Datenbankabfragen gesammelt. Dem Nutzer der Webanwendung wird die Projektnummer, der zuständige Projektbetreuer, das Rechnersystem, der Abrechnungszeitraum, die Art des Kontingents, die Detailinformationen zu jedem Teilnehmer des Projektes und die Kontingenteinheiten angezeigt, einmal monatlich, dann für die ganze Periode und schließlich der prozentuale Wert im Verhältnis zum Gesamtkontingent. Als Detailinformationen eines Benutzers werden sein Account und die monatlich und für eine Periode verbrauchte Anzahl an Kontingenteinheiten extra aufgeschlüsselt. Das bedeutet, dass der Projektleiter sowohl schon zusammengefasste Informationen zu einem Projekt als auch zu einem Benutzer erhält. Die Informationen werden in Form von Tabellen dargestellt. Somit wird die Übersichtlichkeit gewährleistet. Die folgende Abbildung zeigt zu einem Projekt die Ausgabe im Web:

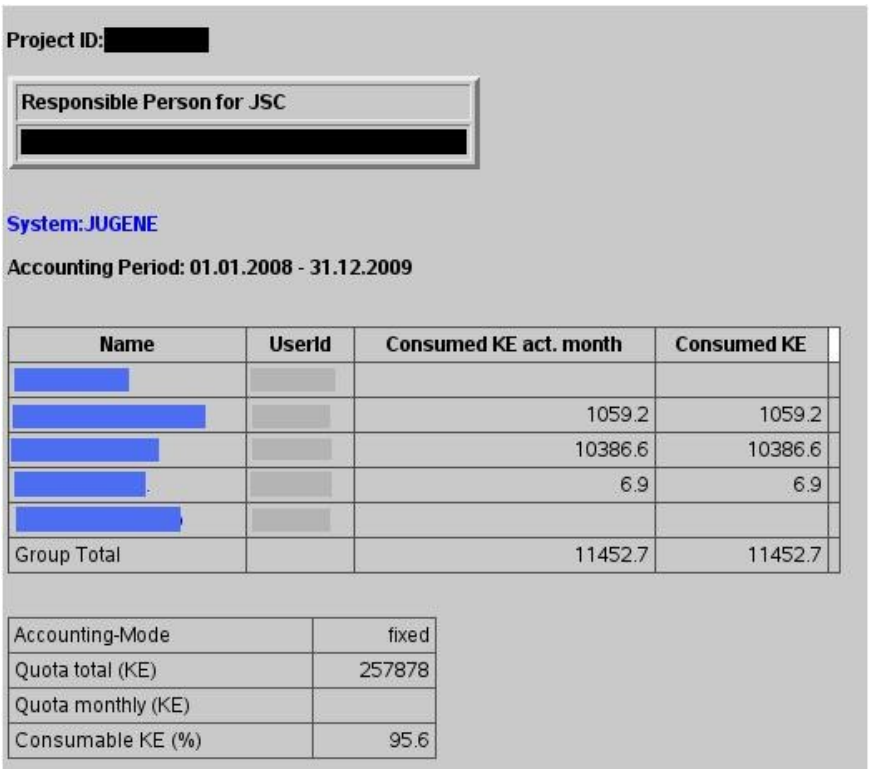

Abbildung 3-8 Kontingentinformationen eines Projektes im Web

Für die Projektleiter besteht zum einen die Möglichkeit ihren Kontingentstand jederzeit über das Web abzufragen. Andererseits werden sie aber auch einmal im Monat per Email informiert.

3.2.5. Jobinformationen auf den Rechnersystemen

Wie unter den Begriffsbestimmungen in Abschnitt 3.1. festgestellt, können die Nutzer über ihren Account auf den zentralen Supercomputer- und Server-Systemen arbeiten. Durch die ausgeführten Programme wird Rechenzeit in Anspruch genommen, die gleichzeitig vom Kontingent abgezogen wird. In diesem Zusammenhang ermöglicht der Kommando-Befehl *q_cpuquota* den Benutzern, jederzeit Informationen zu den genutzten Ressourcen abzufragen. Die Nutzer selbst können mit Hilfe des Kommandos Auskünfte einholen und ihren Verbrauch kontrollieren. Da es sich hier um einen Kommando-Befehl handelt, werden die Daten nicht gespeichert, sondern nur in der Konsole als Text ausgegeben. Durch die Tabellenstruktur ist die Übersichtlichkeit gegeben.

Der Befehl entspricht einem System-Skript, das lokal auf dem Rechnersystem liegt. Somit können nur die Nutzer selbst auf ihre Daten zugreifen. Der Zugriff auf die Informationen der genutzten Ressourcen wird durch das System-Skript durchgeführt. Es fragt über den Unix-Befehl *whoami* die Benutzer-ID ab und identifiziert somit den Benutzer. Durch einen Abgleich mit der Datenbank wird dieser auch verifiziert und erhält nur seine eigenen Daten. Diese werden ebenso von der Datenbank abgerufen.

Das Kommando ermöglicht dem Benutzer durch unterschiedliche Optionen verschiedene Abfragemöglichkeiten. Diese werden in SQL-Abfragen umgewandelt, um die Daten aus der Datenbank zu filtern. Im Wesentlichen erhält der Benutzer Auskunft über:

- den aktuellen Kontingentstatus auf dem gesamten System,
- die Daten zu einem speziellen Job und
- Angaben zu gerechneten Jobs während einem bestimmten Zeitraum oder einer vergangenen Zeitperiode.

Er hat aber auch die Möglichkeit einzelne Attribute abzufragen. Im Einzelnen handelt es sich dabei um Angaben zum Rechnersystem, den Rechnerzugriff, Informationen zur Person und zum Projekt sowie Einzelheiten zum Kontingent bzw. zur Abrechnung. Dem Benutzer wird es somit ermöglicht die Standardausgabe anzupassen und individuelle Informationen nach eigenen Wünschen zusammen zu stellen.

Der Benutzer kann sich auch Informationen zu einem gerechneten Job ansehen. Dazu muss er dem Kommando-Befehl die Jobnummer mitgeben. Das System-Skript erkennt anhand der Parameterliste, welche Informationen gewünscht sind. In diesem Fall werden die Job-Daten von der Datenbank abgefragt und ausgegeben. Die Informationen sind bis zum Vortag verfügbar.

Im Folgenden seien zwei Beispiele angeführt. Das erste in Abbildung 3-9 zeigt den aktuellen Kontingentstatus auf dem gesamten Rechnersystem.

```
q_cpuquota -h jump

============================================================
DV-System:                      JUGENE
Access:                         active
Title:
Name:                           xxxxxx
Firstname:                      xxxxxxx
Organisation/Project:           XXX
E-Mail:                         xxxxxxxxxxxxxxxxx
Tel:                            xxxxxxxx
Loginname:                      xxxxxx
Accounting-Mode:                fixed
Group:                          xxx
Start of Accounting Period:     01.07.08
End of Accounting Period:       30.06.09
State:                          normal
Quota total   (KE):             63310.00 ( 28.00 RD)
Consumed KE (Group):            16891.11 (  7.47 RD)
Consumed KE act. month:          4742.00 (  2.10 RD)
Consumable KE (%):                 73.30
Consumable KE:                  46418.89 ( 20.53 RD)
Consumed KE (User):              5599.21 (  2.48 RD)
```

Abbildung 3-9 *q_cpuquota* für ein Rechnersystem

In der zweiten Abbildung 3-10 werden die Daten zu einem speziellen Job angezeigt.

```
q_cpuquota -j jump01lm.xxxxxx.x

Job-NR           jump01lm.xxxxxx.x
End-Date         20.10.2008 17:09:08
Login-Name       xxxxxx
Walltime (sec)   347
CPU-Time (sec)   15
#cpu  (mode)     64(ST)
#nodes           1
Consumed KE      .925333333333333
Site             fzj
Status           removed
Class            n_short
Node-Usage       not_shared
Jobtype          x
```

Abbildung 3-10 *q_cpuquota* für einen Job

Der Kommando-Befehl kann in gleicher Weise auch von den Projektbetreuern benutzt werden. Er interessiert sich jedoch für die Informationen zu einem bestimmten Projekt oder auch zu den Job-Daten eines speziellen Nutzers. Da der Befehl aber nur Angaben für eine Person individuell macht, muss dieser Mechanismus umgangen werden. Dies geschieht über das Setzen einer globalen Variablen in der Konsole. Das System-Skript prüft dann, ob diese Variable gesetzt ist und liefert die gewünschten Informationen. Somit kann jeder Projektbetreuer für eine fremde Benutzer-ID die Daten abfragen. Diese Ergänzung ist jedoch nur für die Projektbetreuer zugelassen und nicht für jeden beliebigen Benutzer. Das System-Skript vergleicht dazu den Inhalt der Umgebungsvariablen mit der Datenbank. Dort ist in einer Tabelle hinterlegt, welche Personen Projektbetreuer sind. So wird auch nur diesen Personen der Zugriff auf fremde Projektdaten gestattet.

3.3. Problemanalyse

Zum heutigen Zeitpunkt werden für verschiedene Anforderungen unterschiedliche Reports erzeugt. Dabei wird für jede Benutzergruppe eine andere Vorgehensweise zur Erstellung angewandt. Daraus lassen sich die Schwachstellen der derzeitigen Arbeitsweise erkennen.

Der Benutzer hat keinen einheitlichen Zugang zu den Daten. Er muss dafür verschiedene Quellen abfragen. Wie im vorigen Abschnitt erläutert, können die Teilnehmer eines Projektes nur auf dem Rechnersystem selbst ihre Verbräuche ermitteln. Die Projektleiter werden per Email informiert, können aber auch gleichzeitig über das Web ihre Kontingente abfragen. Den Projektbetreuern stehen ebenso mehrere Möglichkeiten zur Verfügung, einerseits über das Web und zum anderen durch den Kommandobefehl *q_cpuquota*.

An die Berichte gelangen die Benutzer über unterschiedliche Authentisierungsmechanismen. Dazu gibt es über den Zugang auf den JSC-Webseiten schon für die Projektleiter mehrere Wege: zum einen über das Projekt-Passwort, aber auch per Email-Verifikation und für Mitarbeiter vom Forschungszentrum zusätzlich über ihr Email-Passwort.

Bei der Erstellung der Reports muss die jeweilige Benutzergruppe berücksichtigt werden, da sie unterschiedliche Berechtigungen haben. Für die Projektbetreuer wird jedoch zurzeit der Zugriff nicht eingeschränkt, sie können auf alle Report-Arten zugreifen.

Für die fünf verschiedenen Arten der Reports existiert jeweils eine andere Erstellung. Zum einen erzeugt der Administrator der Accounting-Software die Auswertungen. Zum anderen werden sie automatisch durch verschiedene Programme angefertigt. Dies erschwert auch die Wartung und Pflege. Ein zusätzlicher Aufwand entsteht für Bedarfsberichte, da diese nur im Einzelfall und somit nicht automatisch erstellt werden können.

Weiterhin wird dem Benutzer nicht die Möglichkeit gegeben, die Reports im Hinblick auf den Inhalt wie auch den Detailierungsgrad selbst zu konfigurieren. Er kann auch nicht über die Darstellungsweise entscheiden. Momentan wird die grafische Darstellung nur bei den monatlichen und jährlichen Verbrauchsstatistiken für das Management genutzt.

Für das Erzeugen der Diagramme wird das Grafikprogramm Gsharp genutzt. Das JSC besitzt dafür eine Lizenz, der Support wird jedoch von der Firma AVS nicht mehr gewährleistet. Falls also Probleme mit der Software auftreten sollten, können die Reports möglicherweise nicht mehr grafisch dargestellt werden.

Zusammenfassend lässt sich also feststellen, dass keine einheitliche Schnittstelle für die Reportgenerierung im JSC existiert, die Nutzer keine eigenen Berichte erstellen können und die grafische Darstellung nur für die Monats- und Jahresauswertungen angewendet wird.

Für das Reporting im JSC eignen sich jedoch nicht die in Unterkapitel 2.6 beschriebenen Reporting-Tools. Mit Hilfe dieser Tools können entweder nur Standardberichte zur Verfügung gestellt werden oder aber der Benutzer muss die Datenbankstruktur genau kennen, um die für den Bericht notwendigen Daten zu ermitteln. Außerdem sind die genannten Reporting-Tools zu komplex aufgebaut. Daher ist es sinnvoll ein eigenes Reporting-Tool für den Einsatz im JSC zu entwickeln.

3.4. Anforderungen und Ziele

Aus den in der Problemanalyse genannten Schwachpunkten werden die Anforderungen und Ziele des zu erstellenden Reporting-Systems abgeleitet. Dazu müssen jedoch einige Randbedingungen beachtet werden. Darauf wird in den nächsten Abschnitten genauer eingegangen.

3.4.1. Anforderungen

Zur Verbesserung der vorhandenen Vorgehensweise werden basierend auf der Analyse in Abschnitt 3.3 folgende funktionale Anforderungen an das neue Reporting-System gestellt:

- Der Zugriff auf die Reporting-Daten muss identitäts- und rollenbasiert beschränkt sein, da nur die Benutzer selbst auf ihre eigenen Daten zugreifen sollen. Zum anderen müssen abhängig von dem jeweiligen Nutzerkreis unterschiedlich detailierte Informationen ausgegeben werden. Insofern kann ein Projektleiter sich alle Informationen zu seinem Projekt ansehen, der Nutzer eines Projektes aber nur die Informationen zu seinen eigenen Programmläufen.
- Den Benutzern soll es ermöglicht werden, die Reports selbst nach ihren gewünschten Anforderungen zu erzeugen. Es sollen aber auch Standardreports zur Verfügung stehen. Durch diese vordefinierten Reports können die Benutzer einfach und schnell grundlegende Informationen erlangen.
- Die erzeugten Reports sollen die Informationen in Tabellenform ausgeben können. Zusätzlich soll eine grafische Darstellung ermöglicht werden. Hierfür sollen insbesondere Linien-, Säulen- und Kreisdiagramme benutzt werden. Aus praktischer Erfahrung haben sich diese Darstellungsarten bewährt.
- Zur Gewährleistung der Übersichtlichkeit soll die Sortierung von Daten zugelassen werden. Weiterhin sollen die Daten aggregiert bzw. gruppiert werden können.
- Die gewünschten Informationen sollen automatisch generiert und durch das System ausgegeben werden. Zum Speichern soll der Report in einem bekannten Dateiformat zum Download angeboten werden.

Zusätzlich zu den funktionalen Anforderungen ergeben sich noch allgemeine Bedingungen. Diese nichtfunktionalen Anforderungen müssen ebenfalls erfüllt werden:

- Das System soll sicher sein, d.h. es muss vor unberechtigtem Zugriff, sowohl versehentlich als auch vorsätzlich, geschützt werden.
- Für das System sollen anerkannte Standards und auch Vorschriften wie z.B. für den Datenschutz berücksichtigt werden.
- Das System soll benutzerfreundlich und verständlich aufgebaut sein, so dass es im praktischen Einsatz von jedem Benutzer verwendet werden kann. Dabei muss auch eine schnelle, einfache und intuitive Bedienbarkeit beachtet werden.
- Das Reporting-System soll modifizierbar und erweiterbar sein. Somit soll der Aufwand der Wartbarkeit gering gehalten werden.

- Um zukünftige Erweiterungen zu erleichtern, ist es sinnvoll, wenn einzelne Programmteile wieder verwendbar sind. Dadurch wird auch die Redundanz von Funktionen umgangen.
- Die Daten sollen in einem angemessenen Zeitrahmen von dem Reporting-System ausgegeben werden, so dass dadurch eine schnellere Auskunft ermöglicht wird.
- Da das Programm von verschiedenen Benutzern innerhalb und außerhalb des Forschungszentrums benutzt werden soll, muss damit auch eine Betriebssystem-unabhängige Lösung gefunden werden.

3.4.2. Randbedingungen und Vorgaben

Da das zu erstellende Reporting-System in eine vorhandene Umgebung integriert werden soll, müssen dazu einige Schnittstellen beachtet werden. Außerdem soll das System auch im produktiven Einsatz verwendet werden, daraus ergeben sich weitere Vorgaben. In der folgenden Aufzählung wird auf die einzelnen Aspekte genauer eingegangen.

- Mit Hilfe des Accounting werden alle notwendigen Daten zu einem Job erfasst, überwacht und verdichtet. Diese werden in dem Oracle-Datenbanksystem gespeichert. Falls sich durch das Accounting Änderungen an der Struktur der Datenbank-Tabellen ergeben, müssen diese auch im Reporting berücksichtigt werden. Somit muss das System die Konfiguration der Datenquelle ermöglichen.
- Das Accounting-System wurde in der Programmiersprache Perl implementiert. Da das Reporting-System auf das Accounting-System aufbauen soll, ist es sinnvoll, die gleiche Programmiersprache zu nutzen.
- In der Oracle-Datenbank sind außerdem die erforderlichen Daten zur Benutzer-Identifikation und teilweise auch der Autorisierung gespeichert. Daher soll die vorhandene Infrastruktur der Datenbank beachtet und darauf aufgebaut werden.
- Das Grafikprogramm Gsharp soll abgelöst werden.

3.4.3. Ziele

Aus den obengenannten Anforderungen ergeben sich folgende Ziele:

Für das Reporting-System muss ein geeigneter *Zugang* für jeden Benutzer geschaffen werden. Dieser soll einheitlich und zentral zugänglich für jede Benutzergruppe sein.

Für die Identifikation und Authentisierung ist zu untersuchen, wie der *Zugriff (Login)* auf das Reporting-System erfolgen soll.

Für die verschiedenen Benutzergruppen soll ein *Rollenkonzept* erstellt werden. Somit wird jeweils einer Benutzergruppe eine definierte Rolle zugeordnet. Bestimmte Informationen können also nur von den jeweils autorisierten Benutzergruppen gesehen werden.

Die *Reporterstellung* soll jederzeit für alle Benutzer ermöglicht werden. Somit können die Berichte nach den eigenen Bedürfnissen in Verwendungszweck, Inhalt und Form verändert werden. Den Benutzern soll darüberhinaus ein Katalog von Standardberichten zur Verfügung

stehen. Neben diesen soll weiterhin die Möglichkeit gegeben sein sich Berichte selbst flexibel und variabel zusammenzustellen.

Die *Präsentation* der gewünschten Informationen soll in optisch aufbereiteter Weise im Web erfolgen. Dafür sollen sowohl Tabellen, Listen wie auch Grafiken verwendet werden können. Diese werden direkt erzeugt und dargestellt.

Das System soll jederzeit zur Verfügung stehen, da es einen sicheren und stabilen Betrieb gewährleisten muss. Weiterhin soll die Anwenderfreundlichkeit garantiert werden, um den Aufwand für die Benutzer so gering wie möglich zu halten.

Für ein neues Reporting-System sollen die Schwachstellen der derzeitigen Vorgehensweise beseitigt werden. Basierend auf den genannten Zielen soll daraus das Konzept des Reporting-Systems erstellt und für den Einsatz im JSC implementiert werden.

Kapitel 4 Design einer Webanwendung

In diesem Kapitel werden zunächst die Grundlagen und wesentlichen Begriffe zum Verständnis einer Webanwendung erklärt. Mit dieser kann ein einheitlicher und zentraler Zugang zu Informationen realisiert werden. Wie schon in der Problemanalyse beschrieben, gibt es derzeit für die einzelnen Benutzergruppen jeweils verschiedene Schnittstellen bzw. Übertragungswege. Anhand der Anwendungsarchitekturen für Webanwendungen wird der spezifische Aufbau für das vorgesehene Reporting-System dargestellt. Danach werden die möglichen Techniken zur Realisierung webbasierter Anwendungen diskutiert und voneinander abgegrenzt. Abschließend wird auf den Ablauf einer geeigneten Technologie und auf die zu gewährleistende Sicherheit der Webanwendung genauer eingegangen.

4.1. Webserver und Webclient

Das World Wide Web besteht aus Webseiten und basiert auf einer Client-Server-Architektur. Zur Kommunikation wird das Protokoll *HTTP* (Hypertext Transfer Protocol) verwendet. Der *Webclient* ist dabei eine Anwendung, die bei jedem Aufruf einer Webseite eine Anfrage (HTTP-Request) an den Webserver sendet. Somit übernimmt der Client bei einer Datenübertragung die Kontaktaufnahme und bestimmt deren Zeitpunkt. Der *Webbrowser* stellt einen besonderen Webclient dar und ist die Schnittstelle zwischen dem Benutzer und dem Web. Der *Webserver* ist ein Server-Programm, welches Ressourcen (z.B. Dateien) zur Verfügung stellt. Für den Webserver spielt es dabei keine Rolle, welche Art von Dateien ausgeliefert werden. Vom Webserver wird die Antwort (HTTP-Response) zurück an den Client gesendet. So wird eine TCP-Verbindung hergestellt, das Dokument übertragen und die Verbindung beendet. Ein Webdokument wird durch das spezielle Adressierungsschema *URL (Uniform Resource Location)* identifiziert und lokalisiert. Sie besteht aus mehreren optionalen Teilen: dem Protokoll, dem Hostnamen des Webserver oder der IP-Adresse, dem Verzeichnisbaum, dem Dateinamen und dem Anker.

Die Grundlage der Darstellung von Informationen im Web ist die Auszeichnungssprache *HTML (Hypertext Markup Language)*. Mit den HTML-Anweisungen, den so genannten Tags, wird die Art und Weise der optischen Darstellung des Inhalts bestimmt. Der Webbrowser erstellt aus diese Anweisungen das Layout der Webseite. HTML-Dokumente können auch Bilder und andere nicht-textuelle Elemente enthalten. Mit Hilfe eines Parsers erkennt der Browser die Elemente und lädt dazu automatisch die zugehörigen Daten. Weiterhin stellt der Webbrowser Services wie z.B. Speichern und Drucken zur Verfügung und kontrolliert Plug-Ins, Komponenten und Hilfsanwendungen. [(Turowski, 2008), (Turau, 1999)]

4.2. Webanwendung

Eine Webanwendung wird nach (Kappel, Pröll, Reich, & Retschitzegger, 2004) wie folgt definiert: Sie „ist ein Softwaresystem, das auf Spezifikationen des World Wide Web Consortium (W3C) beruht und Web-spezifische Ressourcen wie Inhalte und Dienste bereitstellt, die über eine Benutzerschnittstelle, den Webbrowser, verwendet werden“. Die Verarbeitung findet auf dem Webserver statt.

Wenn eine Webanwendung als Zugang für das neu zu erstellende Reporting-System verwendet wird, dann kann eine Vielzahl von Benutzern auch von außerhalb auf die Daten zugreifen. Anwendungen auf einem Computer haben hingegen nur einen beschränkten Benutzerkreis und werden meist lokal gestartet. Bei Webanwendungen sind die teilweise sehr zeitaufwendige Installation von Software oder Programmen auf dem Arbeitsplatzrechner sowie Updates nicht notwendig. Es wird nur einen Webbrowser vorausgesetzt, der ohnehin auf den meisten Systemen vorhanden ist. Außerdem wird ein hoher Grad an Plattformunabhängigkeit erreicht, da sie auf unterschiedlichen Betriebssystemen einsetzbar sind. Im Gegensatz dazu werden normale Anwendungen meist nur für ein Betriebssystem erstellt. Änderungen der Webanwendung müssen nur auf dem Webserver vorgenommen werden. Der Aufwand und die Kosten der Wartung werden somit reduziert. Ein weiterer Vorteil ist die weite Verbreitung von Browsern, dadurch sind Webanwendungen auch auf verschiedenen Endgeräten zugänglich z.B. Mobiltelefone.

Bei Webanwendungen wird zwischen *statischen und dynamischen Webangeboten* unterschieden. Statische Webseiten bestehen nur aus HTML-Code. Im klassischen Fall entspricht eine Webseite genau einer Datei. Die fertigen Dokumente werden auf dem Webserver abgelegt und direkt an den Client ausgeliefert, wenn diese angefordert werden. Dies wird im ersten Teil der Abbildung 4-1 dargestellt.

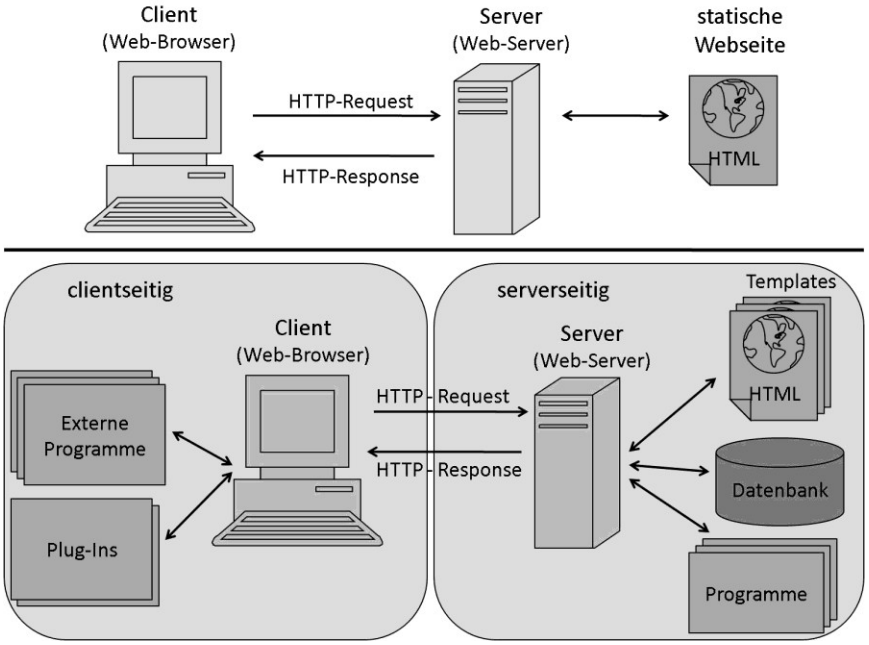

Abbildung 4-1 Funktionsweise von statischen und dynamischen Webanwendungen

Der Nachteil einer solchen Webseite liegt in dem hohen Aufwand der Erstellung und Modifikationen. Um zukünftig alle Informationen darzustellen, wird der Aufbau relativ umständlich oder die Aktualität kann nicht gewährleistet werden. Dynamische Webseiten basieren auf einer Programmiersprache. Erst während des Aufrufes wird der Inhalt der Webseite dynamisch erzeugt. Somit sind Änderungen und Erweiterungen einfach, jedoch ist dazu ein höherer Anfangsaufwand notwendig. Dynamische Webanwendungen ermöglichen Benutzerinteraktionen und es können Softwarekomponenten integriert werden. Im zweiten Teil der Abbildung 4-1 ist das Funktionsprinzip einer dynamischen Webanwendung dargestellt. Es

wird dabei zwischen client- und serverseitigen Tehnologien unterschieden, die in Abschnitt 4.4 näher erläutert werden. [(Kalenborn, 2008), (Hauke, 2008)]

4.3. Architektur

Für Webanwendungen gibt es keine allgemeingültige Architektur. Jedoch werden für die abstrakte Beschreibung der Struktur von Webanwendungen sogenannte *n-Tier-Anwendungsarchitekturen* genutzt. Dabei wird die Architektur in *n* verschiedene Schichten aufgeteilt. Im Gegensatz zum ISO/OSI-Schichtenmodell sind die Schichten nicht horizontal in einem Rechner, sondern vertikal zwischen den Rechnern angeordnet. Ein typisches Beispiel für eine einfache 2-Tier-Architektur ist die Client-Server-Architektur. Das Gesamtsystem besteht aus 2 Komponenten – hier dem Client und dem Server. Die Kommunikation wird in den Schnittstellen festgelegt. Die beiden Komponenten können somit Informationen austauschen, ohne dass jeweils der andere Teil beeinflusst wird. Der Client übernimmt dabei die Präsentation und die Anwendungslogik wird vollständig auf dem Server ausgeführt. Die Ergebnisse der Anwendung werden dann an den Client übermittelt. Die notwendigen Daten werden auf dem Server gehalten und verwaltet. Der Vorteil liegt hier bei den kurzen Entwicklungszeiten und einer schnellen Änderbarkeit. Auf dem Server wird jedoch die ganze Last verteilt und es entsteht eine geringe Skalierbarkeit. Besser eignen sich daher 3-schichtige-Architekturen, die durch das Einführen einer Zwischenschicht entstehen. Dadurch wird die Last gleichmäßiger verteilt und eine höhere Skalierbarkeit erreicht. Je mehr Schichten also eingeführt werden, desto individueller können spezielle Aufgaben optimiert und eine höhere Performance erzielt werden. Dabei ist jedoch der Entwicklungsaufwand und die Kommunikation zwischen den Schichten zu beachten. [(Turowski, 2008), (Kalenborn, 2008)]

Nachdem im Abschnitt 4.2 der Einsatz der Webanwendung begründet wurde, folgt nun die Beschreibung der Struktur. Als Anwendungsarchitektur wird die 3-Tier-Architektur genutzt. Dabei wird die gesamte Funktionalität auf die folgenden 3 Schichten verteilt: Präsentation, Verarbeitung und Persistenz. In Abbildung 4-2 ist die Architektur der Webanwendung zu sehen.

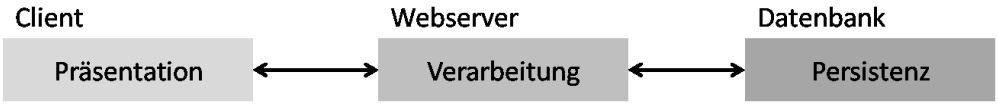

Abbildung 4-2 3-Tier-Architektur der Webanwendung

Bei dieser Architektur ist die Präsentationsschicht für die grafische Aufbereitung und Darstellung der Informationen verantwortlich. Diese kann auf mehreren Clients gleichzeitig erfolgen. Die Präsentation dient außerdem als Benutzerschnittstelle für die Dateneingabe. Auf dem Webserver wird die Anwendungslogik zur Steuerung von Prozessen und Funktionen genutzt. Der Webserver übernimmt somit die Verarbeitung der Webanwendung und steht zwischen der Präsentations- und der Persistenzschicht. Der Client und die Datenbank kommunizieren also nie direkt miteinander. Der Zugriff des Clients auf die Daten wird erst durch die Zwischenschicht, die Verarbeitung, ermöglicht. Das Programm auf dem Webserver wird durch die Präsentationsschicht aufgerufen und damit beginnt dann die Verarbeitung. In der Persistenzschicht werden die Daten für die Webanwendung verwaltet. Diese Schicht wird durch

die Datenbank repräsentiert. Die Datenbank übernimmt dabei das Management, die Speicherung und die Sicherung der Daten.

4.4. Technologien dynamischer Webanwendungen

Webanwendungen können in Abhängigkeit ihres Ausführungsortes in zwei Gruppen aufgeteilt werden: serverseitige und clientseitige Anwendungen. Die folgende Abbildung 4-3 zeigt einen Überblick über die verschiedenen Techniken zur Realisierung webbasierter Anwendungen.

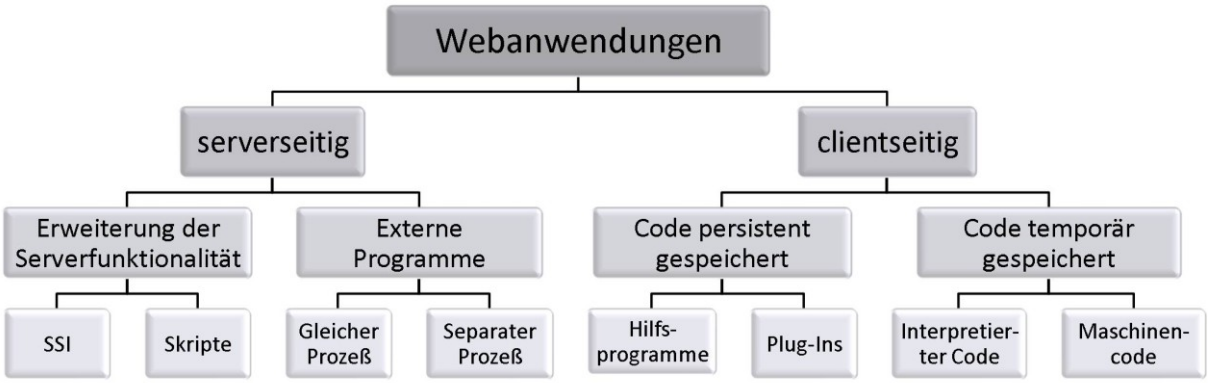

Abbildung 4-3 Klassifikation webbasierter Anwendungen[2]

In diesem Abschnitt werden die Technologien von dynamischen Webanwendungen vorgestellt. In diesem Zusammenhang wird untersucht, welche Technologie sich für das Reporting-System eignet. Gleichzeitig wird überprüft, welcher Lösungsansatz auf die Anforderungen zugeschnitten ist und dabei eine Entscheidung getroffen.

4.4.1. Clientseitige Anwendungen

Um die Kommunikation zwischen dem Webserver und dem Browser zu verringern, können z.B. Eingaben schon auf der Clientseite überprüft werden. Aus diesem Grund wurden clientseitige Anwendungen entwickelt. Das bedeutet jedoch höhere Abhängigkeiten von dem Betriebssystem und der Browser-Software. Zu den clientseitigen Anwendungen zählen sowohl Hilfsprogramme und Plug-Ins wie auch Skripte (JavaScript, VBScript), Java Applets und ActiveX.

Hilfsprogramme und *Plug-Ins* ermöglichen die Verwendung bestimmter Dateitypen wie Multimedia (z.B. Flash), Text-Layout-Formate (z.B. PDF) und verteilte Anwendungen (z.B. Chats). Sie werden bei Bedarf dynamisch geladen und verbleiben auf dem Client. Über definierte Schnittstellen werden sie in den Browser eingebunden. Ein wesentlicher Nachteil ist jedoch die Plattformabhängigkeit von Plug-Ins und Hilfsprogrammen. Für die Umsetzung einer interaktiven Webanwendung ist diese Technologie nicht geeignet.

Demgegenüber stehen Anwendungen, bei denen der Code dynamisch, d.h. nur für die Dauer einer Sitzung, automatisch geladen und danach wieder gelöscht wird. Zu diesen Anwendungen gehören *Skripte* aber auch *Applets*. Dabei können interaktive Inhalte im Browser dargestellt werden. Der Code muss jedoch vor der Ausführung vollständig geladen werden. Daher ist der Quellcode einer Webanwendung einsehbar. Der Programmcode wird direkt an den Client

[2] Aus (Turau, 1999) S. 4

übertragen oder in HTML eingebettet. Bei Skriptsprachen muss der Interpreter vom Benutzer-Browser eingeschaltet sein. Eine wichtige Anwendung von Skripten ist die Validierung der Benutzereingaben. Die Möglichkeiten gehen aber noch viel weiter, dafür bildet das *DOM (Document Object Model)* die Grundlage. Es ist eine allgemeine Beschreibung einer Programmierschnittstelle für Dokumente. Hierfür gibt es jedoch keine einheitliche Unterstützung in den heutigen Browsern. Für Java Applets muss die Java Virtual Machine installiert sein, die auf vielen Clients nicht zwingend vorhanden ist. Applets sind gut geeignet für kleine Client-Dienste oder Applikationen, die nicht aus großen und verschiedenartigen Datenmengen bestehen. Der grundlegende Nachteil ist jedoch, dass für alle erdenklichen Plattformen Versionen bereitgestellt werden müssen, da die Hardware des Clients unbekannt ist. Diese Anwendungen fallen somit aus der Bandbreite der Möglichkeiten heraus.

ActiveX ist die Bezeichnung für ein Softwarekomponenten-Modell von Microsoft. Damit gibt es ActiveX nur für das Windows Betriebssystem. Durch diese enge Verknüpfung mit dem Betriebssystem erleichtert es Angreifern in das System einzudringen, um es zu manipulieren. Aus Sicherheitsgründen unterstützt außer dem Internet Explorer kein anderer Browser ActiveX. Aus diesem Grund kommt diese Technologie für die Webanwendung nicht in Betracht. [(Turau, 1999), (Turowski, 2008)]

Aus den oben angeführten Gründen ist eine clientseitige Webanwendung für das Reporting-System nicht geeignet. Sie dient lediglich der Darstellung von Daten und der Benutzer-interaktion. Im folgenden Abschnitt werden daher serverseitige Anwendungen untersucht.

4.4.2. Serverseitige Anwendungen

Webserver sind für die Bereitstellung der Daten verantwortlich, d.h. das gesamte Programm läuft auf dem Webserver ab. Dadurch wird zwar die Serverlast erhöht, aber die Anwendungslogik geheim gehalten. Serverseitige Anwendungen ermöglichen es, auf Daten zuzugreifen, indem sie die erzeugten HTML-Seiten an den Client schicken. Diese Daten können dabei aus externen Dateien und komplexen Informationssystemen (Datenbanken) stammen oder dynamisch erzeugt werden. Bei serverseitigen Anwendungen wird zwischen externen Programmen, welche vom Server gestartet werden, und Erweiterungen der Serverfunktionalität unterschieden. Zusätzlich wird zwischen Anwendungen differenziert, welche komplette Dokumente erzeugen und solche, die nur Teile in bestehende Dokumente einfügen.

Für die Erweiterung der Serverfunktionalität gibt es verschiedene Techniken. Dabei verändert der Webserver vor dem Abschicken den Inhalt eines Dokumentes, welches er aus dem persistenten Speicher anfordert. Solche Veränderungen werden durch in die Dokumente eingebettete Anweisungen gesteuert. Eine einfache Variante sind *Server Side Includes (SSI)*. Die Bandbreite der Anweisungen ist jedoch relativ klein und hängt vom Webserver ab. Aus diesem Grund kann das Reporting-System selbst nicht mit SSI realisiert werden. Für komplexere Anwendungen werden vollständige Programmiersprachen (z.B. VBScript, JavaScript oder Java) verwendet. Zu den *Skript*-Technologien zählen ASP (Active Server Pages), JSP (Java Server Pages) und PHP (Hypertext Preprocessor oder Personal Home Page). Der Webserver parst die HTML-Dokumente und führt den in den Tags enthaltenen Code aus. Dadurch werden die Dokumente dynamisch konfigurierbar. Die Skript-Tags werden durch die Ergebnisse, den HTML-Anweisungen, ersetzt. Um den Code der Skripte auszuführen, wird jedoch ein Interpreter

benötigt, der über eine Schnittstelle aufgerufen wird. So werden z.B. JSPs mit Hilfe des JSP-Compilers in Servlets kompiliert. Diese Anwendungen hängen daher stark von der Server-Software ab. Mit Hilfe dieser Technik könnte das Reporting-System implementiert werden.

Zu den serverseitigen Anwendungen zählt auch *CGI (Common Gateway Interface)*. Es ist eine Spezifikation, die die Schnittstelle zwischen dem Webserver und den Anwendungen auf diesem Server definiert. Sie bewirkt, dass Webclients angeforderte Ressourcen als Programme auf dem Webserver starten können. Die Programme werden somit auf dem Webserver ausgeführt und nicht zum Client gesendet. Die dynamischen Ressourcen werden vom Webbrowser jedoch auf die gleiche Art und Weise über die URL angefordert wie alle anderen Ressourcen im Web. CGI ermöglicht den Zugriff auf eine Informationsquelle, so dass die Informationen für einen Webclient wie eine Datei auf dem Webserver erscheinen. Die Webseite entsteht durch Programmausgaben, die z.B. HTML-Dokumente oder Bilder erzeugen. Wird eine Anfrage vom Webserver empfangen, so führt dieser das angeforderte Programm aus. Das CGI startet dafür einen neuen Prozess. Diesem Prozess werden die vom Webclient erhaltenen Benutzereingaben mit Hilfe von Umgebungsvariablen oder über Standardeingabe übergeben. Die Umgebungsvariablen werden von dem Programm gelesen und interpretiert. Der Webserver liest die vom Programm geschriebenen Rückgabeinformationen von der Standardausgabe des Prozesses per CGI. Danach wird der Prozess beendet. Bei jedem Aufruf des CGI-Skriptes wird also ein eigener Prozess gestartet. Die Folge davon ist eine relativ geringe Geschwindigkeit von hochfrequentierten Webseiten. Mit der Verwendung von CGI lässt sich jedoch eine komplexe Anwendungslogik in Webseiten integrieren. Sie können in einem hohen Maß dynamisch gestaltet und mit interaktiven Elementen versehen werden. CGI-Anwendungen lassen sich mit vielen Programmiersprachen realisieren. Auch andere serverseitige Sprachen wie PHP laufen über CGI, da die Sicherheitsmaßnahmen mehrstufig aufgebaut sind. Weiterhin ist die Plattformunabhängigkeit auf der Clientseite gegeben. Die typischen Anwendungsgebiete von CGI sind die Verarbeitung von Daten aus Formularen und die Realisierung von Datenbank-Zugriffen. Somit kann CGI ebenfalls für die Webanwendung des Reporting-Systems verwendet werden.

Server APIs sind dynamisch ladbare Bibliotheken. Diese werden einmal geladen und verbleiben im Speicher. Sie arbeiten im gleichen Adressraum wie der Webserver und können somit auf alle verwendeten Ressourcen zugreifen. Jedoch können Fehler in den Anwendungen den Server zum Absturz führen, da sie im gleichen Prozessraum ablaufen. Der Aufwand zur Entwicklung von Server-API-Anwendungen ist oft wesentlich höher als für eine CGI-Anwendung, da auch z.B. die Nebenläufigkeit beachtet werden muss. Ein weiterer Nachteil ist, dass die Anwendung an die Programmiersprache der Server-API gebunden ist. Eine API definiert außerdem nur die Verwendung der Schnittstellen, aber nicht die Realisierung. Sie stellt nur Routinen, Protokolle und Tools für das Erstellen von Anwendungen zur Verfügung. Aus diesen Gründen entfällt die Verwendung von Server APIs für das Reporting-System.

Ein *Servlet* ist ein Server-Applet und damit das Gegenstück zum clientseitigen Applet. Es ist eine auf Java-Technologie basierte Klasse, die auf dem Webserver läuft und üblicherweise HTML-Dokumente erzeugt. Der Server benötigt dazu die Servlet Engine. Diese stellt die Netzwerkdienste zum Empfangen von Anfragen und Senden von Antworten bereit. Die Servlets werden beim ersten Aufruf geladen und verbleiben im Speicher. Die Einbettung von Servlets in HTML-Dokumente erfolgt mit dem Auszeichnungselement <SERVLET>. Der Server parst das

HTML-Dokument, führt referenzierte Servlets aus und ersetzt die Auszeichnungselemente durch entsprechende Ausgaben. Das Servlet selbst erzeugt das gesamte HTML als Ausgabe. Der Lebenszyklus eines Servlets ist genau festgelegt und wird in einem Container verwaltet: Die Servlet-Klasse wird geladen. Danach wird das Servlet-Objekt erzeugt und initialisiert. Es folgt die Verarbeitung der verschiedenen Anforderungen. Am Ende wird das Servlet-Objekt entfernt. Für jeden Benutzer wird somit ein eigenes Objekt im Java-Prozess erstellt. Dadurch wird auch das gleichzeitige Ausführen von mehreren Clientanfragen unterstützt. Servlets eigenen sich ebenfalls für die Anbindung komplexer Standardanwendungen. Sie bieten eine hohe Leistungsfähigkeit und sind unabhängiger als Server APIs. Servlets sind an die Programmiersprache Java gebunden und damit benötigt der Server die Java Virtual Machine. Durch Java wird jedoch die Plattformunabhängigkeit gegeben. Das Erstellen von Servlets ist relativ einfach, da das Servlet API bereits viele Basisklassen enthält. Ein Nachteil ist jedoch, dass der Funktionsumfang der Servlets vom Webserver abhängt. Die Sicherheit wird durch den Servlet-Container gewährleistet, der differenzierte Sicherheitseinstellungen ermöglicht. Servlets werden meist für die Durchführung kleinerer Webdienste eingesetzt. Somit ergibt sich eine weitere Möglichkeit zur Verwendung für das Reporting-System. [(Turau, 1999), (Turowski, 2008), (Hauke, 2008)]

In Tabelle 1 werden noch einmal die wichtigsten Eigenschaften von den geeigneten serverseitigen Technologien gegenübergestellt und zusammengefasst.

	Programmier-aufwand	**Anwendungen**	**Abhängigkeit vom Server**	**Abhängigkeit von der Programmiersprache**
Skripte	mittel	dezidierter Zugriff auf Informationsquellen	hoch	sehr hoch
CGI	mittel	dezidierter Zugriff auf Informationsquellen	keine	fast keine
Servlets	mittel	komplexe oder kooperative Anwendungen	mittel	ausschließlich Java

Tabelle 1 Gegenüberstellung serverseitiger Anwendungen

Nach der Vorstellung der verschiedenen Realisierungsmöglichkeiten muss nun eine Entscheidung für einen Lösungsansatz getroffen werden. Wie in den Randbedingungen (Abschnitt 3.4.2 Randbedingungen und Vorgaben) dargelegt, soll das Reporting-System auf das Accounting-System aufbauen, das in der Programmiersprache Perl implementiert wurde. Es können dadurch bereits existierende Reports mit eingebunden und nach der Implementierung des Reporting-Systems einzelne Programmteile wieder verwendet werden. Dadurch entfallen alle auf anderen Programmiersprachen basierenden Technologien. Insbesondere müsste für Java die Java Virtual Machine installiert werden. CGI-Programme können auch mit der Programmiersprache Perl entwickelt werden, dazu wird das Perl-Modul `CGI.pm` angeboten. Der Vorteil von CGI besteht außerdem darin, dass es ein einfaches und universell verfügbares Mittel zur dynamischen Erzeugung von Webseiten ist. Es müssen also nicht erst bestimmte Softwarekomponenten zusätzlich zum Webserver installiert werden, da CGI bei fast allen Servern mitgeliefert wird. Ein CGI-Programm ist also für das neu zu erstellende Reporting-System gut geeignet und es spricht nichts dagegen die Webanwendung durch die CGI-Schnittstelle zu implementieren.

4.5. Ablauf der Webanwendung

In diesem Abschnitt wird auf den Ablauf der CGI-Webanwendung, welche sich als geeignete Technologie herausgestellt hat, genauer eingegangen. Wenn der Webserver eine Anfrage nach einer statischen Webseite erhält, sucht er das entsprechende HTML-Dokument in seinem Dateisystem und sendet es direkt an den Browser zurück. CGI-Programme werden vom Browser auf die gleiche Art und Weise angefordert. Hierfür schickt der Browser eine HTTP-formatierte Nachricht an den Webserver. Der HTTP-Request enthält die URL, durch die der Server erkennt, welche Ressource er zurückgeben soll. Anstatt den Inhalt der Datei direkt an den Browser zu senden, wird das CGI-Skript ausgeführt. Dafür wird ein neuer Prozess gestartet. Von dem Programm werden die Umgebungsvariablen oder die Standardeingabe gelesen und die Informationen interpretiert. Zur Datenbank wird eine Verbindung aufgebaut und die Abfragen ausgeführt. Die angeforderten Daten werden dann von der Datenbank zurückgeliefert. Die Rückgabe des CGI-Programms erfolgt über die Standardausgabe. Es werden HTML-Tags geschrieben, die letztendlich ein gesamtes HTML-Dokument darstellen. Der Webserver sendet die in der Standardausgabe enthaltenen Daten (HTML-Datei) als HTTP-Response an den Client. Der beschriebene Ablauf wird in Abbildung 4-4 dargestellt.

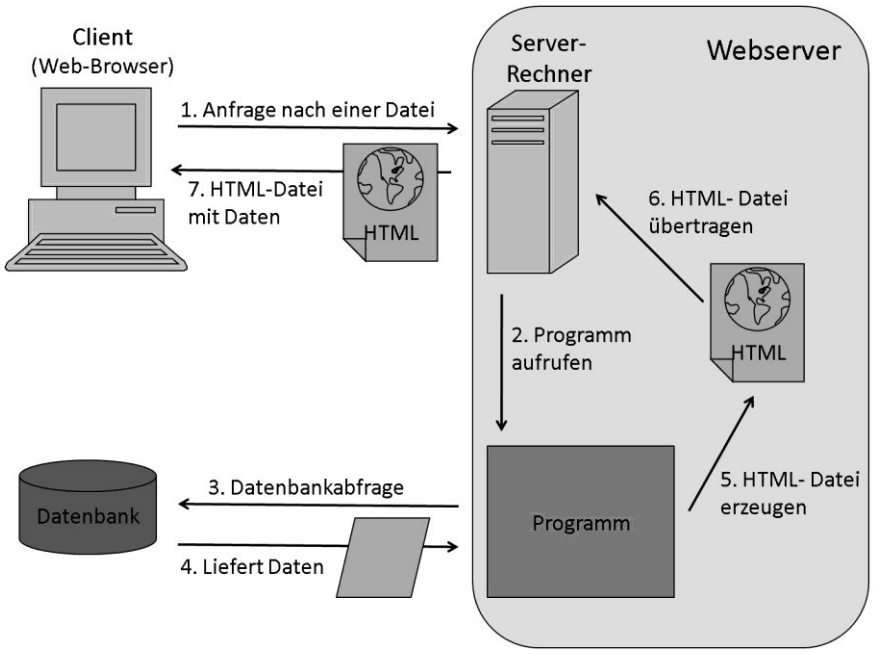

Abbildung 4-4 Ablauf der Webanwendung

4.6. Sicherheit

Webanwendungen sind mit einem höheren Sicherheitsrisiko verbunden als Anwendungen auf Rechnern, welche nicht mit einem Netzwerk verbunden sind. Sicherheitslücken können sich daher direkt auf die Anwendungen auswirken. *Sicherheit* bedeutet in diesem Zusammenhang das Vorhandensein von Integrität, Verbindlichkeit und Vertraulichkeit. Es muss demzufolge Maßnahmen geben, die die Sicherheitsanforderungen erfüllen. Die Ziele von Sicherheitsmaßnahmen sind die Schäden an der Informationsinfrastruktur eines Unternehmens und ihre wirtschaftlichen Folgen zu verhindern. Dazu ist sicherzustellen, dass nur befugte Personen Zugriff auf den Webserver und die Webanwendung haben. Bei dem Zugriff auf die Webanwendung wird die Kontrolle des Ressourcenzugangs durch Authentifizierung und

Autorisierung sichergestellt. Weiterhin muss beachtet werden, dass auf Webservern wertvolle Daten gespeichert werden. Um die Vertraulichkeit zu garantieren, muss die Datenübertragung geschützt werden. Dazu können Nachrichten verschlüsselt werden. Weiterhin muss auch die Datenintegrität gewährleistet werden. Von der Außenwelt kommende Daten müssen auf ihre Validität überprüft werden und dürfen nicht als richtig angenommen werden. Die Verfügbarkeit des Webservers darf auch nicht gefährdet sein, so dass Systemfunktionen nicht beeinträchtigt werden. Ebenso muss der Schutz des internen Netzwerkes vor unberechtigten Zugang durch die Vergabe von Dateirechten gewährleistet werden. Angriffe gegen eine Webanwendung können unter anderem durch die Vermeidung von Sicherheitslücken während der Implementierung abgewehrt werden. [(Turowski, 2008), (Kalenborn, 2008)]

Um die beschriebenen Sicherheitsmaßnahmen für die Webanwendung umzusetzen, wird hier zunächst auf die Kommunikation zwischen dem Webserver und dem Client eingegangen. Da es sich um sensible Daten handelt, muss die Datenübertragung geschützt werden. HTTP garantiert keine sichere Übertragung. Aus diesem Grund wird *HTTPS (Hypertext Transfer Protocol Secure)* verwendet. HTTPS ist ein auf Basis von *SSL/TLS (Secure Socket Layer/ Transport Layer Security)* verschlüsseltes HTTP. SSL/TLS stellt einen sicheren Kanal für die HTTP-Kommunikation zur Verfügung, der zwischen der Anwendung (HTTP) und dem Transport (TCP) angeordnet ist. Sie dient zur Verschlüsselung und zur Authentifizierung der Kommunikation zwischen dem Webserver und dem Browser. Ohne Verschlüsselung sind die übertragenen Daten für jeden, der Zugang zum entsprechenden Netz hat, als Klartext lesbar. Außerdem bietet es Sicherheitsmechanismen, die das Belauschen der Kommunikation und anderer Angriffe auf die Vertraulichkeit der Daten verhindern. Es stellt das einzige Verschlüsselungsverfahren dar, das ohne gesonderte Softwareinstallation auf allen Computern unterstützt wird. Außerdem ist mit SSL/TLS eine geschützte Identifikation und Authentifizierung der Kommunikationspartner möglich. Dabei kann SSL/TLS unterscheiden zwischen anonymen Verbindungen, der Server-Authentifizierung und der Client-Server-Authentifizierung. Der Standard-Port für HTTPS-Verbindungen ist 443. Es ist in jeden modernen Webbrowser integriert und daran erkennbar, dass die URL mit dem Schema https:// beginnt. SSL/TLS verwendet dazu digitale Zertifikate, die nach dem weit verbreiteten Standard ITU X.509 aufgebaut (siehe dazu auch Abschnitt 5.1.4 Benutzerzertifikate). Beim Aufruf der URL wird zunächst vom Webserver ein Schlüssel-Zertifikat, das den öffentlichen Verschlüsselungsschlüssel des Webservers enthält, zurück gesendet. Der Webserver kann sich somit gegenüber dem Client identifizieren. Der Browser überprüft das Zertifikat auf Gültigkeit und generiert einen symmetrischen Sitzungsschlüssel und schickt diesen mit dem öffentlichen Schlüssel des Webservers verschlüsselt an den Webserver. Nachdem beide Seiten den Sitzungsschlüssel kennen, erfolgt die weitere Kommunikation symmetrisch verschlüsselt über SSL/TLS. Der Webserver muss dabei nicht erfahren, mit wem er wirklich kommuniziert. SSL/TLS ist ein Beispiel für die Anwendung von hybrider Kryptographie. In der Abbildung 5-1 in Abschnitt 5.1.4 wird der Ablauf der verschlüsselten Kommunikation zwischen dem Client und dem Server dargestellt. Auf weitere Sicherheitsmaßnahmen wie die Kontrolle des Ressourcenzugangs wird im nachfolgenden Kapitel im Rahmen der Konzeptionierung genauer eingegangen. [(Guelich, Gundavaram, & Birznieks, 2001), (SoftEd Systems, 2009), (Kersken, 2006)]

Kapitel 5 Konzeptionierung des Reporting-Systems

Dieses Kapitel dient zur Erläuterung des Konzeptes für das Reporting-System. Dafür werden hier die Ziele aus Abschnitt 3.4.3 einzeln aufgegriffen und im Rahmen der Konzepterstellung genauer beleuchtet. Zunächst wird dazu die Zugangskontrolle der Webanwendung vorgestellt. Im Rahmen der Zugriffskontrolle wird das Rollenkonzept des Reporting-Systems erläutert. Anschließend folgt eine detailierte Darstellung zur Reporterstellung. Im letzten Abschnitt dieses Kapitels geht es um die Präsentation der Reportdaten.

5.1. Zugangskontrolle

Durch die Webanwendung wird ein zentral erreichbarer Zugang für alle Benutzer geschaffen. Somit kann prinzipiell jeder auf das Reporting-System zugreifen. Dabei muss natürlich beachtet werden, dass nicht jeder die Berechtigung hat, das System zu nutzen. Dafür wird eine Zugangskontrolle eingeführt. Darunter ist ein System oder auch Verfahren zu verstehen, das die Identitäten seiner Kommunikationspartner erfragt, prüft und nur mit berechtigten Partnern weiter kommuniziert. Die Zugangskontrolle verhindert so die unbefugte Inanspruchnahme der Betriebsmittel. Ein System kann einen Menschen daran erkennen, wer er ist, was er hat und was er weiß. Auf diese Weise wird die eindeutige Bestimmung der Identität (*Identifikation*) vorgenommen. Daher ist es möglich, nicht nur den Zugang zu beschränken, sondern gleichzeitig den Benutzer zu identifizieren. Für das Reporting-System spielt dieser Zusammenhang eine wesentliche Rolle, da somit auch die Ressourcen dem einzelnen Benutzer richtig zugeordnet werden können.

Für die Zugangskontrolle gibt es unterschiedliche Verfahren, die jedoch nach demselben Prinzip verlaufen: Zuerst muss die Identität des Kommunikationspartners erfragt werden. Der Benutzer muss sich also gegenüber dem Webserver authentisieren. Dabei können auch eindeutige Kennungen (Alias) zugelassen sein. Als *Authentisierung* wird der Nachweis der eigenen Identität bezeichnet. Nur bei erfolgreicher Prüfung der wahren Identität eines Benutzers (*Authentifizierung*) wird die Kommunikation fortgesetzt. Dazu kann ein Nutzer anhand von biometrischen Merkmalen z.B. dem Fingerabdruck identifiziert werden (Wer ist der Benutzer?). Es ist aber auch möglich, den Benutzer durch eine Kombination aus Benutzernamen und Passwort zu authentisieren (Was weiß er?). Eine dritte Variante zur Erkennung einer Person wird durch den Besitz eines Nutzers wie z.B. dem Ausweis oder auch Schlüssel ermöglicht (Was hat er?). Die Prüfung der Identität kann entweder nur am Anfang oder dauerhaft während der gesamten Kommunikation erfolgen. [(Federrath, 2008), (Pfitzmann, 2000)]

Die Authentisierung über biometrische Merkmale ist für das Reporting-System nicht angemessen. Die Entwicklung ist noch nicht soweit ausgereift, sodass sie nicht zuverlässig eingesetzt werden kann. Für die Erkennung der biometrischen Merkmale sind zusätzlich spezielle technische Vorrichtungen notwendig. Außerdem können der Datenschutz und die Persönlichkeitsrechte verletzt werden. Die beiden anderen Methoden – Authentisierung mit dem Wissen oder dem Besitz eines Benutzers – können jedoch für das Reporting-System eingesetzt werden. Dazu werden in den folgenden Abschnitten einzelne Beispiele erläutert, die zu diesen Verfahren gehören.

5.1.1. Benutzername und Passwort

Die Verwendung von Benutzernamen und Passwort ist die klassische Form der Benutzer-Authentisierung und zählt zur Methode der Authentisierung mit Wissen. Hierbei wird die Kenntnis einer Information, die nur dem Benutzer bekannt ist, als Nachweis verwendet.

Der Benutzer meldet sich dazu mit seinem Benutzernamen oder einer anderen eindeutigen Kennung (BenutzerID) an und authentisiert sich durch sein Passwort. Diese Daten werden verschlüsselt an den Webserver gesendet. Damit das Reporting-System den Benutzer authentifizieren kann, müssen die Daten verifiziert werden.

Wie bereits in Kapitel 3 erwähnt, existiert für die Projektleiter ein Passwort zur Abfrage der Projektdaten, das den Nutzern des Projektes jedoch nicht bekannt ist. Dieses Passwort ist zentral in der Datenbank gespeichert. Der Zugang des Reporting-Systems wird in diesem Fall durch die Angabe des Projektes mit dem dazugehörigen Passwort ermöglicht. Dabei muss jedoch schon bei der Anmeldung ein spezielles Projekt ausgewählt werden. Dies ist jedoch nicht wünschenswert, da an dieser Stelle noch keine Einschränkungen vorgenommen werden sollen. Zudem werden die verschiedenen Rollen eines Benutzers nicht berücksichtigt.

Eine andere Möglichkeit für die Authentisierung wäre, die BenutzerID zu verwenden. Dazu muss für die Projektleiter und Betreuer, die keine ID für ein Projekt besitzen, zusätzlich eine ID für das Reporting-System angelegt werden. Für die Benutzer könnte das Passwort genutzt werden, mit dem sie auf ein Rechnersystem zugreifen. Die Passwörter werden in einem Verzeichnisdienst verwaltet. Dieser setzt sich aus einer X.500-Datenbank und dem dafür entwickelten Zugriffsprotokoll LDAP (Lightweight Directory Access Protocol) zusammen. In der X.500-Datenbank kann jegliche Art von Informationen z.B. auch die Konfiguration und Administration verschiedener Anwendungen erfasst werden. Der Verzeichnisdienst liegt jedoch nur lokal auf den Supercomputern. Somit müsste entweder eine Verbindung direkt zum Reporting-System hergestellt werden oder zur Oracle-Datenbank. Weiterhin sind in dem LDAP-Verzeichnisdienst nur die Benutzer-Passwörter gespeichert und damit werden die Projektleiter und die Betreuer nicht erfasst. Die Zuordnung der Rollen ist bereits in der Oracle-Datenbank abgelegt und müsste daher nicht noch zusätzlich im LDAP gespeichert werden. Der Verzeichnisdienst auf den Supercomputern wird also lediglich für die Benutzer-Authentisierung der Supercomputer verwendet.

Für das Reporting-System müsste also ein eigenes Passwort angelegt werden. Es ist dabei aber nicht unbedingt notwendig, ein neues Passwort zu erstellen. Dazu können die vorhandenen Passwörter der Benutzer und Projektleiter verwendet werden. Diese werden entweder in der Datenbank oder direkt auf dem Webserver gespeichert, um den Benutzer authentifizieren zu können.

Bei der Verwendung von Passwörtern sind allerdings folgende Gesichtspunkte zu beachten: Die Benutzer müssen sich durch die Vielzahl der mittlerweile existierenden Anwendungen eine Reihe von BenutzerIDs und Passwörtern merken. Dies führt dazu, dass die Benutzer aus Bequemlichkeit die gleichen Passwörter verwenden oder die Passwörter leicht zugänglich direkt am Arbeitsplatz aufgeschrieben oder abgespeichert werden. Meist sind die Passwörter einfach gehalten, um sie sich leicht merken zu können. Dabei ist darauf zu achten, dass die Passwörter durch das Ausprobieren aller Möglichkeiten nicht gehackt werden. Besser eignen sich komplexe

Passwörter, die zusätzlich regelmäßig geändert werden müssen. Sie sind jedoch weniger benutzerfreundlich und können dazu führen, dass das Reporting-System weniger genutzt wird. Die Sicherheit der Webanwendung liegt somit auch in der Hand des Benutzers, da dieser die Richtlinien für Passwörter beachten und einhalten muss. Der wesentliche Vorteil der Benutzer-Authentifizierung über Benutzername und Passwort ist die einfache Anwendbarkeit. Sie eignet sich für die Zugangskontrolle von Bereichen mit einzuschränkenden Inhalten. Für die Berichte des Reporting-Systems werden allerdings sensible Daten benutzt. Dafür ist die Anwendung von Passwörtern allein, heute nicht ausreichend sicher. Dieses Verfahren wird deshalb für die Zugangskontrolle nicht angewendet. Falls zukünftig der Einsatz von Passwörtern gewünscht wird, muss ein umfassenderes Konzept erarbeitet werden, das auch die genannten Gesichtspunkte berücksichtigt. [(Todt & Bauer, 2004), (Federrath, 2008)]

5.1.2. Einmalpasswort

Bei der klassischen Authentisierung wird ein statisches Passwort verwendet. Im Gegensatz dazu ist das Einmalpasswort nur für einen einzigen Vorgang gültig und kann kein zweites Mal verwendet werden. Somit können Angreifer mitgelesene Passwörter nicht für eigene Zwecke missbrauchen. In den meisten Fällen setzt sich die Authentifizierung aus einer statischen PIN und einem dynamischen Einmalpasswort zusammen. Die Schwierigkeit ist dabei die Synchronisation, denn sowohl der Server als auch der Benutzer müssen wissen, welches Kennwort für einen bestimmten Anmeldevorgang gültig ist. Dazu werden Kennwortlisten oder auch Kennwortgeneratoren eingesetzt. Im ersten Fall müssen die vorgefertigten Listen (z.B. TAN-Listen) auf beiden Seiten hinterlegt werden. Die Listen werden entweder sequentiell abgearbeitet oder ein zufälliger noch nicht benutzter Wert wird ausgewählt. Bei den Kennwortgeneratoren wird durch einen speziellen Algorithmus jeweils ein aktuelles Passwort generiert. Dabei wird aber nicht der Algorithmus, sondern nur das Ergebnis übertragen. Dieses wird entweder zeit- oder ereignisgesteuert generiert. In beiden Fällen wird die Berechnung des Kennwortes sowohl vom Client als auch vom Server durchgeführt. Der Server muss zum Abgleich jedoch einen gewissen Toleranzbereich beachten, da z.B. die eingebaute Uhr nicht hundertprozentig exakt funktioniert. Ein Beispiel für einen zeitgesteuerten Generator ist das SecurID Token, es handelt sich dabei um eine Art Schlüsselanhänger mit LCD-Anzeige. Diese Verfahren bieten einen sehr hohen Schutz gegenüber den herkömmlichen Authentifizierungssystemen. Die Kosten für zusätzliche Anschaffung der Hardware und der Server-Software sind jedoch sehr hoch. Der Standard des Authentifizierungsverfahrens ist außerdem nicht frei zugänglich. Für das Reporting-System sind diese Verfahren zu aufwendig und zu kostenintensiv. Weiterhin muss der Benutzer die Kennwortliste oder Generator immer mitführen, deswegen werden sie nicht eingesetzt. [(Todt & Bauer, 2004), (Wikipedia, Die freie Enzyklopädie, 2009)]

5.1.3. Email

Für Authentifizierung per Email kann zur Authentisierung die Email-Adresse genutzt werden. Bei dieser Methode handelt es sich um die Authentisierung mit Besitz, denn der Benutzer verfügt über eine Email-Adresse. Um sich bei der Webanwendung anzumelden, muss der Benutzer seine Email-Adresse angeben. Diese muss dann vom Reporting-System überprüft werden. Dieser Mechanismus wird bereits für die Projektleiter und auch die Projektbetreuer eingesetzt, falls sie

ihr Projektpasswort nicht eingeben wollen. Die Emailadresse ist eindeutig einer Person zugeordnet und eignet sich daher besser als der Benutzername. Die Adressen sind auch zentral in der Oracle-Datenbank gespeichert, sodass die Webanwendung jederzeit darauf zugreifen kann. Dazu muss kontrolliert werden, ob sie in der Datenbank vorhanden ist. Danach wird an diese Email-Adresse die URL der Webanwendung gesendet. Die Email muss von dem Benutzer erst abgeholt werden, um auf die URL zugreifen zu können. Weiterhin muss die URL genau auf die jeweilige Person zugeschnitten sein. Sie kann jedoch einfach verändert werden. Mittlerweile werden Email-Adressen überall im Web veröffentlich. Somit kann grundsätzlich jeder, eine andere Adresse angeben. Der eigentliche Benutzer erhält dann eine Email und wundert sich, da er selbst die Adresse nicht angegeben hat. Er wird dadurch belästigt und stuft die Email als Spam ein. Die Emails können aber auch umgeleitet bzw. abgefangen werden. Damit wird die URL an eine andere Person gesendet und dieser der Zugriff auf die Anwendung gewährt. Diese Vorgehensweise der Authentifizierung ist für das Reporting-System nicht ausreichend sicher und kann eine Person nicht eindeutig identifizieren. Die Emailadresse sollte also nicht allein, sondern immer in Kombination mit einem anderen Authentisierungsmerkmal genutzt werden, um die Benutzer eindeutig zu identifizieren. Zudem sollen die Benutzer direkt auf die Webanwendung zugreifen können und nicht erst über einen Umweg an die Informationen gelangen. Aus diesen Gründen wird dieses Verfahren nicht eingesetzt.

5.1.4. Benutzerzertifikate

Der Einsatz von Benutzerzertifikaten zählt ebenfalls zur Authentisierung mit Besitz, da dem Benutzer das digitale Zertifikat gehört. Zertifikate enthalten in der Regel folgende Informationen: den Namen des Ausstellers, Ausgaberegeln und –verfahren, die Gültigkeitsdauer, den öffentlichen Schlüssel, den Namen des Eigentümers und weitere Informationen zum Eigentümer, Angaben zum zulässigen Anwendungs- und Geltungsbereich sowie die digitale Signatur des Ausstellers über alle Informationen. Die Gültigkeit ist auf den im Zertifikat festgelegten Zeitraum begrenzt. Typische Anwendungsbereiche von digitalen Zertifikaten sind elektronische Signaturen zum Schutz von Emails. Weiterhin dienen sie zur Sicherheit in Netzwerkprotokollen (HTTPS/SSL).

Zur besseren Veranschaulichung ist in der Abbildung A-2 im Anhang die Text-Darstellung eines nach X.509v3 (Version 3) aufgebauten digitalen Zertifikats dargestellt. X.509 ist ein Standard für digitale Zertifikate und bietet eine hohe Sicherheit.

Für die Erstellung eines digitalen Zertifikates muss ein Schlüsselpaar aus einem öffentlichen und privaten Schlüssel generiert werden. Die Zertifizierung bezieht sich auf den öffentlichen Schlüssel und den Zusammenhang zwischen ihm und dem Teilnehmer. Eine Zertifizierungsstelle überprüft zunächst die Identität des Antragstellers anhand von Ausweispapieren. Zertifizierungsstellen werden auch als *Certification Authority (CA)* bezeichnet. Danach wird das eigentliche Zertifikat erstellt, indem über die Angaben des Zertifikates eine Prüfsumme gebildet und mit dem privaten Schlüssel verschlüsselt wird. Die verschlüsselte Prüfsumme und weitere Angaben bilden die digitale Signatur. Diese Signatur stellt den Zusammenhang der Identität des Benutzers mit dem öffentlichen Schlüssel her und kann der handschriftlichen Unterschrift gleichgestellt werden. Die wesentlichen Bestandteile dieses nun offiziellen Dokumentes sind also der öffentliche Schlüssel, die Angabe des Gültigkeitszeitraumes und eine Angabe darüber, zu wem der öffentliche Schlüssel gehört (digitale Signatur). Mit diesen

Bestandteilen kann aber noch keine Berechtigung gegeben werden. Das bedeutet, dass der Besitz eines Zertifikates den Benutzer nicht eindeutig identifiziert. Erst durch den privaten Schlüssel, der zu dem im Zertifikat beglaubigten öffentlichen Schlüssel passt, wird der Benutzer eindeutig identifiziert. Somit muss der private Schlüssel geschützt und geheim gehalten werden.

Das Zertifikat wird zusätzlich zum privaten Schlüssel durch die Zertifizierungshierarchie abgesichert. An der Spitze stehen wenige Wurzelinstanzen, die die Zertifikate der CA sind und als vertrauenswürdig gelten. In der Hierarchie stellen die höheren Instanzen jeweils die Zertifikate für niedrigere Zertifizierungshierarchien aus. Diese zusätzliche Absicherung wird dort gebraucht, wo sich jemand gegenüber einer fremden Instanz ausweisen muss. Die CA bildet somit das Bindeglied zwischen den Kommunikationspartnern, dem beide Partner vertrauen und dessen Wurzelzertifikat sie besitzen.

Zertifikate gewähren in Verbindung mit HTTPS die Authentisierung (Nachweis, dass die Nachricht vom Inhaber des Zertifikates stammt), die Integrität (Nachweis der Unverfälschtheit von Daten) und die Vertraulichkeit (Verhinderung des Zugriffs von unberechtigten Dritten). Außerdem kann durch die digitale Signatur der Benutzer identifiziert werden. Im Zusammenhang mit dem privaten Schlüssel wird die Identität eindeutig nachgewiesen. [(Richter, 2005), (Schoenebeck & Sczimarowsky, 2002)]

Digitale Zertifikate werden bereits im JSC eingesetzt. Sie werden im Rahmen des verteilten Höchstleistungsrechnen (Computational Science) bei der Benutzung von UNICORE[3] (Uniform Interface to Computing Resources) angewendet und ebenso im DEISA-Projekt[4] (Distributed European Infrastructure for Supercomputing). Durch die Zertifikate wird der sichere Zugriff auf die verfügbaren Ressourcen der Supercomputerzentren gewährleistet. Weiterhin besitzen viele Benutzer ein Zertifikat, um ihre Emails durch digitale Signaturen zu schützen. Dafür befindet sich im JSC eine Registrierungsstelle (FZJ-RA), die die Zertifizierung des öffentlichen Schlüssels initiiert. Sie ist für das Bearbeiten von Zertifikats- und Sperranträgen, die Archivierung der Antragsformulare, die Beratung und den Support verantwortlich. Die potentiellen Nutzer des Reporting-Systems besitzen also schon ein Zertifikat und der Umgang ist ihnen vertraut. Zukünftig soll auch der Einsatz von Zertifikaten im JSC erweitert werden.

Da bereits für die sichere Übertragung mit HTTPS Zertifikate verwendet werden (siehe Abschnitt 4.6), kann für die Authentifizierung die technische Umsetzung genutzt werden. Aufgrund der genannten Punkte wird für das Reporting-System die Authentisierung mit Benutzerzertifikaten eingesetzt.

Für die Anmeldung am Reporting-System muss das Zertifikat im Browser zur Verfügung stehen. Das Client-Zertifikat des Benutzers wird an den Webserver übertragen und verifiziert. Dazu wird zunächst überprüft, ob die digitale Signatur der CA in dem Zertifikat des Kommunikationspartners korrekt ist. Der Webserver übernimmt somit die Überprüfung der Gültigkeit über den privaten Schlüssel und die Zertifizierungshierarchie. Die Server-Authentifizierung gegenüber dem Client und die Verschlüsselung der Kommunikation werden bereits durch die Verwendung von HTTPS realisiert. Der genaue Ablauf wird in der Darstellung Abbildung 5-1 veranschaulicht.

[3] http://www.unicore.eu
[4] http://www.deisa.eu

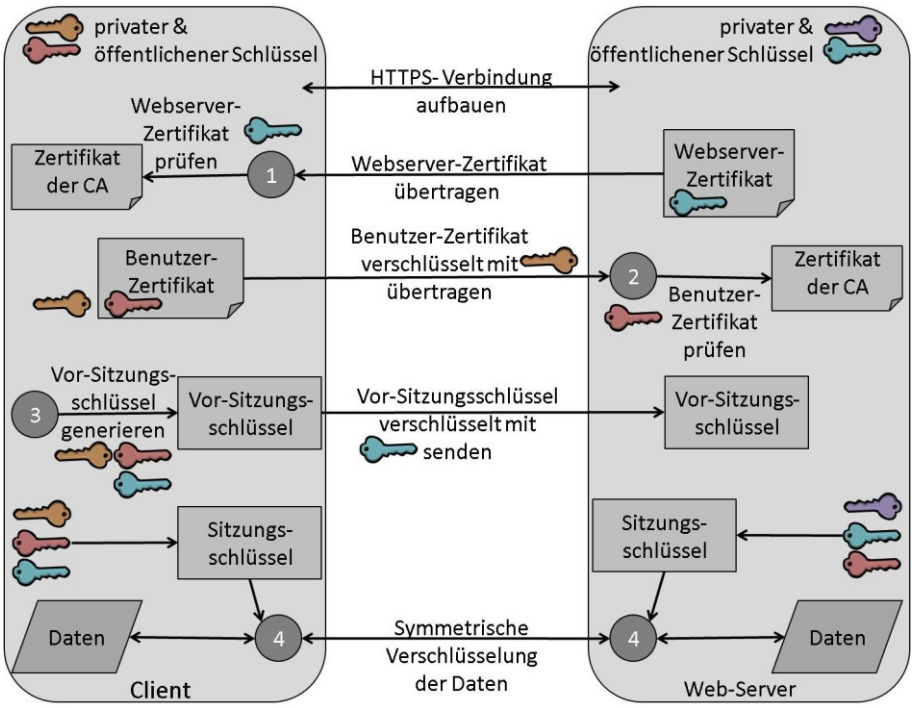

Abbildung 5-1 Ablauf der SSL-Verschlüsselung

Vom Reporting-System muss jetzt geprüft werden, ob der Inhaber des Zertifikates für die Nutzung dieser Anwendung autorisiert ist. Dies wird durch einen Abgleich mit der Oracle-Datenbank realisiert. Der allgemeine Aufbau dieser Datenbank wird im nachfolgenden Abschnitt 5.2.2 erläutert. In der Tabelle *d_pers* sind die Angaben zu allen Benutzern gespeichert. Jedoch ist das Zertifikat eines Benutzers nicht in der Datenbank abgelegt. Eine genaue Übersicht ist in der Abbildung 5-5 dargestellt. Als erste Möglichkeit könnte deswegen der Namen des Zertifikatinhabers mit dem Benutzernamen in der Datenbank verglichen werden. Dieser Abgleich ist jedoch nicht eindeutig, da ein Name auch doppelt in der Datenbank vorkommen kann. Die Verwendung der Emailadresse würde sich dadurch besser eignen. Diese ist in der Datenbank-Tabelle für jeden Benutzer eindeutig, da eine Emailadresse nur einmal vergeben wird. Im Zertifikat wird die Emailadresse jedoch nur optional angegeben. Daher ist dieser Vergleich auch nicht möglich. Das Zertifikat enthält aber einen zur Person eindeutigen Schlüssel, dieser wird als Distinguished Name (DN) bezeichnet. Dieser Name wird im Zertifikat wie folgt dargestellt:

```
Subject: C=DE, O=Forschungszentrum Juelich GmbH, OU=JSC, CN=Cornelia Geyer
```

Abbildung 5-2 Distinguished Name eines Zertifikates

Dazu ist es erforderlich, diesen Schlüssel zentral in der Datenbank-Tabelle *d_pers* zu speichern. In der Tabelle wird dafür eine neue Spalte angelegt werden. Somit kann der DN auch für andere Anwendungen eingesetzt werden. Die Daten für die Authentifizierung werden also nicht auf dem Webserver gespeichert, dies erhöht zusätzlich die Sicherheit.

5.2. Zugriffskontrolle

Durch die Zugangskontrolle wird der Benutzer eindeutig identifiziert. Dadurch ist es möglich entsprechende Rollen und somit die Rechte dem Benutzer zu zuordnen. Die *Zugriffskontrolle* setzt also die Zugangskontrolle voraus. Für die Zugriffskontrolle müssen zunächst Rollen

definiert werden. Jede Rolle hat vordefinierte Rechte und dadurch können nur bestimmte Operationen ausgeführt werden. Die Rechtevergabe selbst wird *Autorisierung* genannt. Somit wird genau festgelegt, wer welche Berechtigungen hat und wer unter welchen Umständen was darf. Dazu wird eine geeignete Auswahl an Operationen zur Verfügung gestellt, jedoch den berechtigten Benutzern nicht alles erlaubt. Abhängig von der jeweiligen Benutzergruppe wird der Zugriff auf die Ressourcen dynamisch eingeschränkt. Durch die Zugangs- und Zugriffskontrolle können weiterhin individuell angepasste Sichten und Menüs eines Benutzers dargestellt werden.

5.2.1. Rollenkonzept

Aus Datenschutzgründen muss der Zugriff für die verschiedenen Benutzer auf seine Daten eingeschränkt werden. Bestimmte Informationen dürfen also nur von den jeweils autorisierten Benutzergruppen gesehen werden. Dafür wird einer Benutzergruppe eine Rolle zugeordnet. Je nach Rollenzuordnung des Benutzers erteilt oder sperrt das Reporting-System dann den Zugriff auf die Daten. Eine Rolle wird dabei je nach Aufgabenbereich definiert. Dadurch müssen die Rechte nicht für jeden Benutzer einzeln festgelegt werden. Statt den Benutzern die Rechte direkt zu zuweisen, wird eine Rolle definiert, die dann vielen Benutzern zugeordnet werden kann. Demzufolge gibt es also wesentlich weniger Rollen als Benutzer. Dies erleichtert die Verwaltung der Rechte für das Reporting-System, da bei Änderungen der Struktur nur die Rechte der Benutzerrolle angepasst werden müssen. Es werden zunächst vier Rollen angelegt:

- Die Nutzer der Rolle *User* dürfen ausschließlich die Daten zu ihrer genutzten Rechenleistung einsehen. Insbesondere sollen alle Informationen angezeigt werden, die zu den gerechneten Jobs gehören.
- Dem Projektleiter wird die Rolle *Project Manager* zugeordnet. Dadurch erhält der Projektleiter Einblick in die Daten seines Projektes. Er ist aber auch befugt, die Daten der Benutzer seines Projektes zu überprüfen.
- Wie bereits in dem Abschnitt 3.1 beschrieben, ist einem Projekt auch ein Projektbetreuer zugewiesen. Der Betreuer muss daher ebenfalls einen Einblick in die Daten des zugeordneten Projektes haben. Die Rolle des Projektbetreuers wird *Project Adviser* genannt. Diese Rolle besitzt zurzeit die gleichen Rechten wie der *Project Manager*.
- Die *Administrator*–Rolle darf uneingeschränkt auf sämtliche verfügbare Informationen zugreifen. Hierbei kann es sich um den Administrator des Reporting-Systems handeln oder auch die Person, die die monatlichen und jährlichen Berichte für das Management erstellt.

Somit sind diese Rollen für das Reporting-System eindeutig definiert. Das Rollenkonzept basiert dabei auf den existierenden Verantwortlichkeiten und Zuständigkeiten einer Person. Daher werden die Rollen auch in dieser Weise aufgeteilt. Falls sich neue Aufgabenbereiche ergeben, kann das Rollenkonzept jederzeit erweitert werden. Es handelt sich hierbei um ein hierarchisches Rollenkonzept, in dem eine untergeordnete Rolle in einer übergeordneten Rolle enthalten ist. Der Benutzer des Reporting-Systems kann gleichzeitig mehrere Rollen haben z.B. *User* des Projektes A und *Project Manager* des Projektes B. Wenn die Laufzeit des Projektes B endet, wird dem Benutzer nur die Rolle des *Project Managers* und die damit verbundenen übergeordneten Rechte entzogen. Dieses Rollenkonzept wird in der Abbildung 5-3 dargestellt.

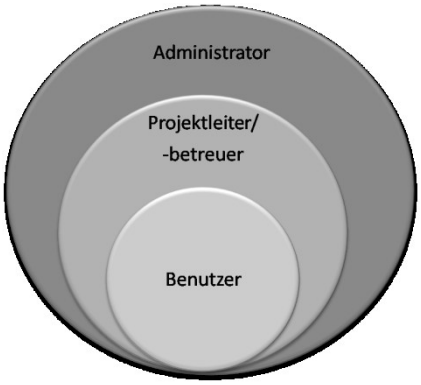

Abbildung 5-3 Rollenkonzept

Ein anonymer Zugang zum Testen des Reporting-Systems z.B. durch ein Gast-Benutzerkonto ist nicht sinnvoll. Dieser Benutzer kann demnach nicht identifiziert und ihm können keinerlei Rechte zugeordnet werden. Dadurch besitzt ein anonymer Benutzer keine Ressourcen. In die Datenbank müssten dafür fiktive Testdaten eingefügt werden. Da sich diese sich aber im produktiven Einsatz befindet, kann dies nicht ermöglicht werden. Für die Erstellung eines Reports stehen daher keine Daten zur Verfügung und die Nutzung bleibt ergebnislos.

Wie bereits in Abschnitt 4.3 beschrieben, sind die für das Reporting-System notwendigen Benutzerdaten in der Dispatch-Datenbank gespeichert, so auch die Rollen der Benutzer. Um den Benutzern ihre Rechte zu zuordnen, werden durch Datenbankabfragen die entsprechenden Rollen ermittelt. Dazu wird im folgenden Abschnitt die Datenbankstruktur erläutert.

5.2.2. Dispatch-Datenbanktabellen

Im JSC wird das Datenbanksystem Oracle genutzt. Es basiert auf dem relationalen Datenbankmodell, das Tausende von Transaktionen pro Sekunde verarbeiten kann. In dem Modell werden die Datenobjekte und deren Beziehungen in Tabellen gespeichert. Der konzeptuelle Entwurf der Dispatch-Datenbanktabellen wird in der Abbildung 5-4 gezeigt. Diese Tabellen werden vom Dispatch verwaltet und gepflegt. Die Daten sind nach ihrer Zugehörigkeit getrennt in den Tabellen abgelegt.

Anhand des dargestellten Entity-Relationship-Diagramms sind die einzelnen Beziehungen der Tabellen erkennbar. Die Elemente des Diagramms sind die Entitäten (Rechtecke), die Attribute (Ovale) und die Beziehungen (Rauten) zwischen den Entitäten. Dabei sind jeweils nur die für das Reporting-System relevanten Attribute dargestellt. In einem relationalen Datenmodell werden die Entitäten als Tabelle umgesetzt. Die Attribute legen fest, welche Spalten in der Tabelle gespeichert werden. Eine Zeile der Tabelle entspricht einem Tupel und steht für einen Datensatz. Das Schlüsselattribut (unterstrichen) dient dabei zur eindeutigen Identifikation eines Datensatzes. Wie in der Darstellung ersichtlich, haben die Beziehungen unterschiedliche Wertigkeiten. Eine *1:n* Beziehung besagt, dass einer Entität maximal *n* Entitäten gegenüberstehen, *n* kann dabei aber auch Null sein. Die Entität Person entspricht der Tabelle *d_pers* und stellt sämtliche benutzerbezogenen Informationen bereit. Der Personenschlüssel kennzeichnet eine Person, also genau einen Datensatz in der Tabelle. Eine Person kann mehrere Accounts besitzen. Die Accounts sind in der Tabelle *d_uid_host* gespeichert. Dabei werden für das Reporting-System die Spalte Loginname und das dazugehörige Rechnersystem benutzt.

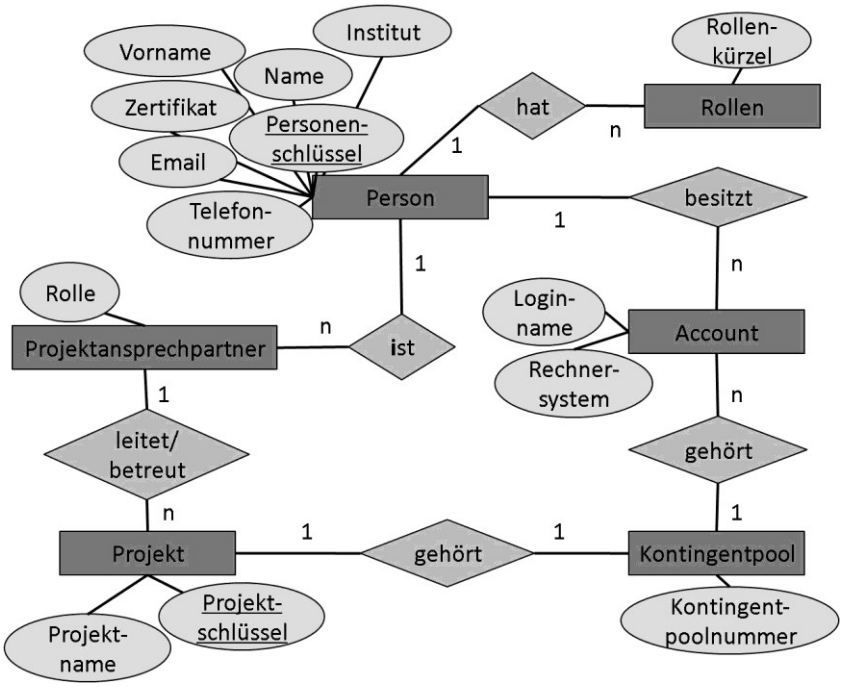

Abbildung 5-4 Entity-Relationship-Diagramm der Dispatch-Datenbanktabellen

Falls eine Person Projektleiter oder Betreuer ist, wird dieser Zusammenhang als Rolle in der Tabelle *d_org_anspr* gespeichert. Der Projektansprechpartner leitet oder betreut ein Projekt. Ein Projekt wird in der Datenbanktabelle *d_org* erfasst und hat einen Projektnamen und einen eindeutigen Projektschlüssel. Ein Projekt gehört zu einem Kontingentpool *d_cont_pool*, über den ein Projekt abgerechnet wird. Einem Kontingentpool können dann mehrere Accounts zugeordnet werden. Um die Rolle des Administrators zu erfassen, wurde die Tabelle *d_jurep_rollen* erstellt und in der Datenbank ergänzt. Der Administrator kann nicht wie die Rollen Projektleiter und Betreuer in der Tabelle *d_org_anspr* gespeichert werden, da sich diese Rollen direkt auf ein bestimmtes Projekt beziehen. Für den Administrator kann und soll jedoch dieser Bezug nicht hergestellt werden, da er umfassende Rechte besitzen muss. Außerdem können in der Tabelle *d_jurep_rollen* auch weitere Rollen hinzugefügt werden, falls sich zu einem späteren Zeitpunkt zusätzliche Rollen ergeben, die unabhängig von einem Projekt sind.

Die Beziehungen zwischen den Entitäten werden durch das Übernehmen des Schlüssels in die andere Tabelle realisiert. In der Abbildung 5-5 wird dies durch die Pfeile erkenntlich. Die für das Reporting-System relevanten Spalten einer Tabelle sind markiert. In der Darstellung werden die einzelnen Tabellenbeschreibungen angezeigt. In der ersten Spalte steht der Spaltenname einer Tabelle. Die *Null?*-Spalte gibt an, ob diese Spalte in der Tabelle Daten enthalten muss (NOT NULL) oder nicht. Beim Einfügen eines neuen Datensatzes müssen diese Felder einen Wert besitzen. Die dritte Spalte gibt den Datentyp an. In den Dispatch-Datenbanktabellen werden die folgenden Datentypen verwendet:

- VARCHAR2(n) – Hierbei handelt es sich um eine Zeichenkette variabler Länge mit maximal n Zeichen.
- NUMBER(n) – Dieser Datentyp ist eine Festkommazahl mit n Stellen.
- DATE – Das Datum wird mit interner Zeitangabe gesichert.

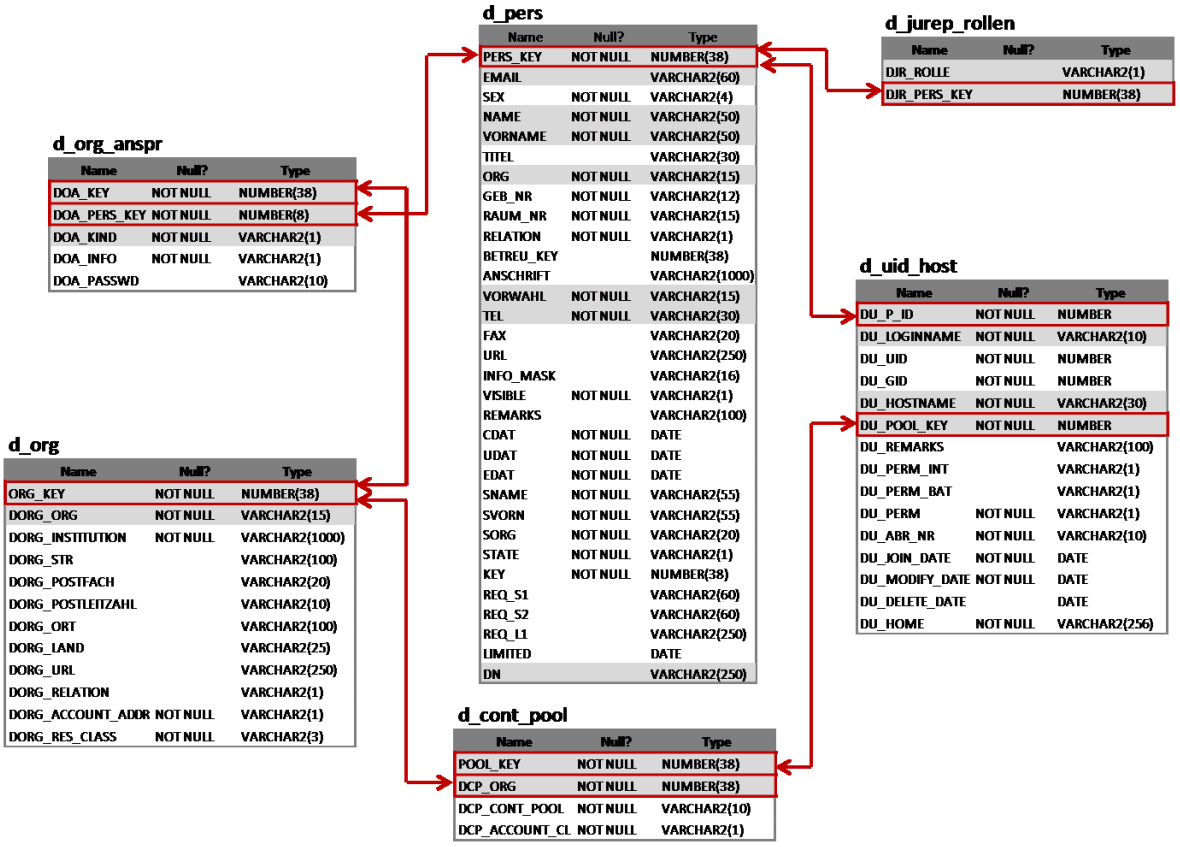

Abbildung 5-5 Aufbau der Dispatch-Datenbanktabellen

Für das Reporting-System wird zur Abfrage der Daten die Datenbanksprache *SQL (Structured Query Language)* verwendet. Die Sprache dient auch der Definition und Manipulation von Daten in relationalen Datenbanken. Es ist ein internationaler Standard und die gemeinsame Sprache verschiedener Datenbanksysteme. SQL wird auch vom Datenbanksystem Oracle unterstützt.

5.3. Reporterstellung

Wie in Kapitel 2.2 beschrieben, müssen zunächst die gesammelten Daten bereitgestellt werden, die zum Erstellen der Berichte notwendig sind. Dazu muss auf eine Datenbasis zugegriffen werden, die die Informationen zu den Jobs auf den verschiedenen Rechnersystemen enthält. Von dem Reporting-System werden die Daten des Accounting verwendet. Diese Informationen werden durch das Accounting-System von den genutzten Rechnersystemen gesammelt und in der zentralen Accounting-Datenbank gespeichert. Somit kann der Nachweis über die erbrachte Rechenleistung geführt werden. Der folgende Abschnitt gibt dazu einen Einblick in das Accounting und dessen Datenbanktabellen.

5.3.1. Accounting

Die für das Accounting erforderlichen Angaben werden auf den verschiedenen Rechnersystemen in den Log-Dateien gespeichert. Es handelt sich dabei um die Informationen zu den Jobs wie der Benutzeraccount, die Zeitpunkte an denen ein Job gestartet und beendet wurde, die verbrauchten CPU-Zeiten und die Speicherbelegung. Diese Angaben beziehen sich auf einen Job, der durch

seine JobID eindeutig gekennzeichnet wird. In der Abbildung 5-6 wird ein Beispiel-Datensatz des Linux-Cluster-Systems SoftComp angezeigt.

```
jobidXXX  090326221414 090326222041 090326222521 useridX     group
jobname_xxxxxx     serial          110          280       150268       444636
1  1     0  N  PBS
```

Abbildung 5-6 Log-Datensatz vom Linux Cluster SoftComp

Die Log-Daten werden während eines ganzen Tages vom Betriebssystem auf den Rechnersystemen erzeugt und in Dateien gesammelt. Die Daten werden am darauffolgenden Tag durch das Accounting-System von dem Rechnersystem in die Datenbank übertragen. Daraus ergibt sich für das Reporting-System, dass nur die Daten bis zum Vortag ausgegeben werden können. Außerdem werden beim Übertragen der Daten in die Datenbank diese nicht 1:1 übernommen. Es werden zusätzliche Angaben zu einem Benutzer gespeichert und einzelne Informationen zu einem Job zusammengefasst und aufbereitet. Die detailierten Jobinformationen eines Tages werden in der Datenbank zudem monatlich verdichtet. Daher existiert immer zu einer detaillierten Tabelle eine monatliche Tabelle. Die Daten sind in den Accounting-Datenbanktabellen also in einem unterschiedlichen Detailierungsgrad gespeichert. Dabei ist auch zu beachten, das die detaillierten Daten in der Tabelle maximal drei Monate gesichert werden, während die Zusammengefassten bis zu zwei Jahre gespeichert sind. Die einzelnen Daten werden aus Speicherplatzgründen und zur besseren Handhabung in komprimierter Form hinterlegt.

Derzeit existieren im JSC vier verschiedene Rechnersysteme, für die ein Accounting durchgeführt wird. Die Daten dieser Rechnersysteme werden alle nach ihrer Zugehörigkeit getrennt in den unterschiedlichen Tabellen gespeichert. Dies ist durch unterschiedliche Log-Daten der verschiedenen Rechnersysteme begründet. In der folgenden Abbildung 5-7 werden die existierenden Datenbanktabellen zu dem jeweiligen Rechnersystem aufgelistet. Dabei gehört zu jedem Tabellennamen jeweils eine Tages- und Monatstabelle.

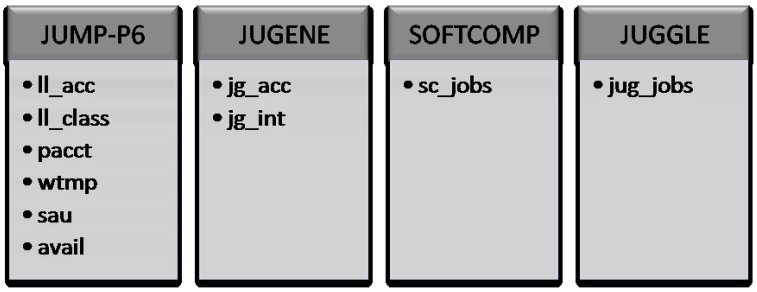

Abbildung 5-7 Accounting-Datenbanktabellen

Wie aus der Darstellung ersichtlich wird, gibt es für die Supercomputer JUMP-P6 und JUGENE mehrere Tabellen. In diesen Tabellen werden auch andere Informationen nicht nur auf Job-Ebene abgebildet. Die einzelnen Datenbanktabellen werden in der folgenden Aufzählung vorgestellt. Um einen Überblick zu den einzelnen Daten zu erhalten, wird die erste aufgeführte Tabelle ausführlicher erläutert.

- Die Tabellen *ll_acc_j* und *ll_acc_m* enthalten die Jobinformationen vom LoadLeveler für den Supercomputer JUMP. Der LoadLeveler[5] ist ein Jobscheduler, über den die

[5] http://www-03.ibm.com/systems/clusters/software/loadleveler/index.html

Rechenjobs im Batchbetrieb ablaufen. Er bietet ausführliche Informationen für das Accounting. In der Tabelle *ll_acc_j* werden die Daten täglich in detaillierter Form gespeichert. Zu dem Tabelleninhalt zählen die Benutzerinformation, Angaben zum Accounting, die Laufzeiten, bestimmte Zeitpunkte, Speicherbelegungen, Daten zu den genutzten Knoten, einzelne Eigenschaften und zusätzliche Angaben jeweils bezogen auf einen Job. Die Benutzerinformationen umfassen die Organisation, den Benutzernamen, den Benutzeraccount und die Benutzergruppe. Dadurch wird ein Job eindeutig seinem Benutzer zugeordnet. Für das Accounting werden in den Datenbanktabellen eine eindeutige Accounting-Nummer, der Kontingentpool, zu dem das Kontingent des Projektes gehört, und die abzurechnenden Accounting-Einheiten des Jobs gesichert. Bei den Job-Laufzeiten wird unterschieden zwischen der gesamten Laufzeit vom Eingangs- bis zum Enddatum, der CPU-Laufzeit, der eigentlichen Laufzeit des Jobs und der Systemzeit. Dabei wird für das Accounting die gesamte Laufzeit abgerechnet, die sich aus den anderen genannten Zeiten zusammensetzt. Die bestimmten Zeitpunkte werden für die Laufzeiten gespeichert, um nachvollziehen zu können, wann der Job auf dem System eingegangen ist, wann er gestartet und beendet wurde. Für die Speicherbelegung wird jeweils der minimal und maximal gebrauchte Speicherbedarf erfasst. Für einen Job werden ebenfalls die Anzahl der genutzten Knoten, CPUs und Hosts, der Threadmodus und die Auslastung der Knoten gesammelt. Jobs werden teilweise in einzelne Arbeitsschritte (steps) unterteilt. Dazu werden in den einzelnen Eigenschaften eines Jobs die eindeutige StepID, die Klasse, der Status und der jeweilige Typ angezeigt. In der Tabelle *ll_acc_m* werden die Jobs mit gleichen Eigenschaften monatlich zusammengefasst. In der Tabelle *ll_class_m* werden die Jobinformationen nach verschiedenen Jobklassen komprimiert, in die die Jobs nach ihrer Größe aufgeteilt werden.

- In den Tabellen *pacct_d* und *pacct_m* sind die Kommando-Informationen von dem Login-Knoten des JUMP abgelegt. Dabei werden Angaben zu den ausgeführten Kommandos ähnlich zu den Jobinformationen abgelegt.
- Die Anzahl der Logins werden in den Tabellen *wtmp_d* und *wtmp_m* gespeichert. Dazu werden jeweils das Startdatum und die Dauer eines Logins zu einem Benutzeraccount gesammelt.
- In den Tabellen *sau_d* und *sau_m* wird die Knotenauslastung des Rechnersystems erfasst. Zu den einzelnen Knoten und zu den jeweiligen Zeitpunkten werden die Prozentangaben zu den Auslastungen auf der CPU-, der Benutzer-, der System- und der I/O- Ebene gespeichert.
- Die Informationen zu einer Wartung oder aber zu einem Ausfall des Supercomputers JUMP werden in den Tabellen *avail_s* und *avail_m* festgehalten. Dabei werden der betroffene Knoten, die Start- und Endzeit, die jeweilige Dauer und die Art des Ausfalls gespeichert
- Die Jobinformationen vom LoadLeveler für JUGENE werden in den Tabellen *jg_acc_j* und *jg_acc_m* aufgeführt, diese sind analog zu den Jobinformationen von JUMP, unterscheiden sich aber in einigen rechnersystemspezifischen Merkmalen.

- Die Tabellen *jg_int_i* und *jg_int_m* enthalten die Wartungsinformationen für den Supercomputer JUGENE. Diese Tabellen umfassen die gleichen Angaben wie die Tabellen *avail_s* und *avail_m,* zusätzlich wird noch die Partition angegeben.
- Für das Cluster-System SoftComp werden die Jobinformationen von Torque in die Tabellen *sc_jobs_j* und *sc_jobs_m* übertragen. Torque[6] (Terascale Open-Source Resource and Queue Manager) wird als Batch-System genutzt und organisiert die Verarbeitung der Jobs auf den Compute-Knoten. In den Datenbanktabellen werden dazu die Benutzerinformationen, die Zeitpunkte der Jobs, die Laufzeiten, der Speicherverbrauch, die Accounting-Informationen und zusätzliche Angaben gespeichert. In der Abbildung 5-8 werden die SoftComp-Datenbanktabellen dargestellt. Anhand dieses Beispiels sollen die einzelnen Jobinformationen für ein Rechnersystem veranschaulicht werden. Die markierten Zeilen in der Tabelle *sc_jobs_j* sind direkt auf einen Job bezogen und werden nicht in die Tabelle *sc_jobs_m* übernommen. Dafür gibt es in der komprimierten Tabelle gesonderte Felder, um spezielle Eigenschaften eines Jobs zu bewahren. Ein Beispiel dafür ist das Feld `nojobs`, das die Anzahl der Jobs in einem zusammengefassten Datensatz enthält.
- In den Tabellen *jug_jobs_j* und *jug_jobs_m* werden die Jobinformationen für JUGGLE gesichert, die ebenfalls von Torque generiert sind. Diese Tabellen sind ähnlich wie die SoftComp-Datenbanktabellen aufgebaut.

sc_jobs_j

Name	Null?	Type
ORG	NOT NULL	CHAR(15)
ACCNO	NOT NULL	CHAR(10)
USERNAME	NOT NULL	CHAR(45)
GROUPID	NOT NULL	CHAR(8)
CONTPOOL	NOT NULL	CHAR(9)
A_CLASS	NOT NULL	CHAR(1)
R_CLASS	NOT NULL	CHAR(3)
JOBID	NOT NULL	NUMBER(38)
SUBDATE		DATE
STARTDATE		DATE
ENDDATE	NOT NULL	DATE
USERID	NOT NULL	VARCHAR2(8)
JOBNAME	NOT NULL	VARCHAR2(15)
QUEUE	NOT NULL	VARCHAR2(10)
CPUTIME		FLOAT(126)
WALLTIME		FLOAT(126)
MEM		FLOAT(126)
VMEM		FLOAT(126)
ERRVAL		NUMBER(38)
VE		FLOAT(126)
SEPJOB	NOT NULL	VARCHAR2(1)
CPUS		NUMBER(38)

sc_jobs_m

Name	Null?	Type
ORG	NOT NULL	CHAR(15)
ACCNO	NOT NULL	CHAR(10)
USERNAME	NOT NULL	CHAR(45)
GROUPID	NOT NULL	CHAR(8)
CONTPOOL	NOT NULL	CHAR(9)
A_CLASS	NOT NULL	CHAR(1)
R_CLASS	NOT NULL	CHAR(3)
ENDDATE	NOT NULL	DATE
USERID	NOT NULL	VARCHAR2(8)
QUEUE	NOT NULL	VARCHAR2(10)
CPUTIME		FLOAT(126)
WALLTIME		FLOAT(126)
MEM		FLOAT(126)
VMEM		FLOAT(126)
NOJOBS		NUMBER(38)
ERRVAL		NUMBER(38)
VE		FLOAT(126)
SEPJOB	NOT NULL	VARCHAR2(1)
CPUS		NUMBER(38)
MAX_CPUS		NUMBER(38)
WALLCPU		FLOAT(126)

Abbildung 5-8 Datenbanktabellen vom Linux Cluster SoftComp

5.3.2. Flexible Erstellung eines Reports

Aus dem Abschnitt 5.2.1 Rollenkonzept wird deutlich, dass das Reporting-System für verschiedene Benutzergruppen konzipiert ist. Der Nutzer eines Projektes interessiert sich für die Laufzeiten seiner gerechneten Jobs. Der Projektleiter wiederum möchte sich über sein Projekt informieren. Andererseits soll der Administrator des Reporting-Systems Monatsauswertungen

[6] http://www.clusterresources.com/products/torque-resource-manager.php

für das Management anfertigen können. Durch eine flexible Reporterstellung ist der Benutzer in der Lage, seinen Bericht nach den eigenen Bedürfnissen und Zielsetzungen selbst zusammenzustellen. Dabei sollen die Berichte in Verwendungszweck, Inhalt und Form variabel sein. In konkreten Situationen z.B. zur Entscheidungsfindung sind die Berichte auch kurzfristig erstellbar. Um einen Report zu erzeugen, müssen folgende Schritte umgesetzt werden:

1. Bereitstellen der Informationen
2. Auswahl der Informationen für einen Report – Konfiguration der Abfrage
3. Abruf der Daten und Generieren des Reports.

Bereitstellen der Informationen

Für die flexible Erstellung muss der Benutzer wissen, welche Informationen überhaupt zur Verfügung stehen. Aus dem vorherigen Abschnitt 5.3.1 wird deutlich, dass in der Datenbank vielfältige Informationen gespeichert sind. Es muss also zunächst ein Konzept für das Bereitstellen der Daten und der Zugriff auf diese entwickelt werden. Eine erste Möglichkeit wäre, die verschiedenen Tabellen in der Datenbank zu einer gesamten Tabelle für alle Systeme zusammenzufassen. Dazu müsste zunächst definiert werden, welche Daten die Tabelle enthalten soll. Einerseits können diejenigen Spalten ausgewählt werden, die in jeder Tabelle von sämtlichen Rechnersystemen enthalten sind. Dadurch wird jedoch nur ein Minimum an Daten definiert und die auswählbaren Informationen werden eingeschränkt. Eine solche Einschränkung ist jedoch nicht gewünscht. Der Benutzer soll alle seine gespeicherten Daten einsehen können. Andererseits können auch alle Daten in die Tabelle eingefügt werden. Es steht also ein Maximum an Daten zur Verfügung. Dabei ist zu beachten, dass von den Rechnersystemen unterschiedliche Daten in den Accounting-Tabellen gespeichert werden. In einigen Tabellen sind manche Spalten nicht angelegt, diese würden in der Tabelle dann leer bleiben. Bei der Abfrage dieser Spalten können dann keine Daten ausgegeben werden. Der Benutzer weiß in diesem Fall jedoch nicht, welche Felder eines Datensatzes zum jeweiligen System definiert sind. Dieser Ansatz ist somit nicht geeignet.

Die Daten des Accounting werden also aus den rechnersystemspezifischen Tabellen oder Sichten genutzt. Dabei ist zunächst zu überlegen, in welche Informationen einem Benutzer der Einblick gewährt werden darf. Ein Nutzer eines Projektes soll z.B. keine genauen Informationen zur Wartung oder zum Ausfall eines Rechnersystems erhalten. Da bereits verschiedene Rollen existieren, müssen nur die Rechte zu den jeweiligen Tabellen definiert werden. Dazu wird ein XML-Schema entworfen.

XML (eXtensible Markup Language) ist eine Metasprache für die Beschreibung und Auszeichnung von Dokumenten und eine Teilmenge von *SGML (Standard Generalized Markup Language)*. XML-Dokumente enthalten nur Text und keine binären Daten. Damit sind sie visuell lesbar und können mit jedem Texteditor bearbeitet werden. XML ist plattformunabhängig und maschinenlesbar, so dass es für den Einsatz im WWW angepasst ist. XML ermöglicht neben der Beschreibung auch den Austausch nahezu aller Arten von Daten zwischen Computersystemen. Die besondere Bedeutung von XML für das Internet liegt darin, dass sich dadurch eine Vielzahl von Auszeichnungssprachen, wie z.B. auch SVG, für spezielle Datenaustauschzwecke definieren lassen. XML-Dokumente sind ähnlich definiert wie HTML-Dokumente, da das Konzept von Tags und Attributen äquivalent zu dem von HTML ist. Im

Gegensatz zu HTML ist allerdings kein Standard-Satz von Elementen vorgegeben, sondern die Tags und Attribute sind in XML beliebig definierbar. Ein Grundgedanke von XML ist, dass die Struktur, der Inhalt und die Formatierung voneinander getrennt sind. Dazu kann ein XML-Schema oder eine *DTD (Document Type Definition)* verwendet werden. Die Strukturbeschreibung dient dann zur strukturellen Validierung der Daten. Es besteht die Möglichkeit, eine individuelle Auszeichnungssprache durch die Definition eigener Tags zu bilden. Dazu können die Informationen über die Bedeutung des Inhalts in den Tag-Namen abgelegt werden. Somit sind auch Datenstrukturen frei definierbar. Die XML-Datei ist hierarchisch aufgebaut und entspricht im logischen Aufbau einer Baumstruktur. Mit Hilfe der Verschachtelung von Tags kann somit das Text-Dokument strukturiert werden.

In der Abbildung 5-9 wird das XML-Dokument `tabellenrechte.xml` gezeigt, in dem die Accounting-Datenbanktabellen jeder einzelnen Rolle zugewiesen werden.

```
<authorisations>
	<role name="user">
		<system name="jump">
			<detailed>pacct_d, ll_acc_j</detailed>
			<combined>pacct_m, ll_acc_m</combined>
		</system>
		<system name="softcomp">
			<detailed>sc_jobs_j</detailed>
			<combined>sc_jobs_m</combined>
		</system>
		<system name="juggle">
			<detailed>jug_jobs_j</detailed>
			<combined>jug_jobs_m</combined>
		</system>
		<system name="jugene">
			<detailed>jg_acc_j</detailed>
			<combined>jg_acc_m</combined>
		</system>
	</role>
	<role name="manager">...</role>
	<role name="adviser">...</role>
	<role name="admin">
		<system name="jump">
			<detailed>pacct_d, ll_acc_j, wtmp_d, sau_d,
			    avail_s</detailed>
			<combined>pacct_m, ll_acc_m, wtmp_m, ll_class_m, sau_m,
			     avail_m</combined>
		</system>
		<system name="softcomp"> ... </system>
		<system name="juggle"> ... </system>
		<system name="jugene">
			<detailed>jg_acc_j, jg_int_i</detailed>
			<combined>jg_acc_m, jg_int_m</combined>
		</system>
	</role>
</authorisations>
```

Abbildung 5-9 XML-Definition der Tabellenrechte

Es beginnt mit einem Prolog, in dem die XML-Deklaration steht. Weiterhin kann eine XML-Datei aus einem oder mehreren Elementen bestehen, die durch Tags abgebildet werden. Um

sicherzustellen, dass ein XML-Dokument verarbeitet werden kann, muss es *wohlgeformt* sein. Dafür gibt es gewisse Mindestanforderungen, die befolgt werden müssen:

- Das Dokument besitzt das Wurzelelement `<authorisations>`, welches den weiteren Inhalt des Dokumentes umschließt.
- Die Elemente können Attribute besitzen. In der Datei hat das Element `role` das Attribut `name`. Es besteht aus einem Namen `name`, dem ein Wert in Anführungszeichen `"user"` zugewiesen werden muss. Ein Element darf nicht mehrere Attribute mit gleichem Namen besitzen.
- Alle Elemente mit Inhalt müssen durch ein Start-Tag und dem dazugehörigen End-Tag umschlossen werden, z.B. `<detailed>sc_jobs_j</detailed>`.
- Die Elemente dürfen einander nicht überschneiden, sie dürfen nur auf einer Ebene verschachtelt werden. [(Turowski, 2008), (Bos & Fischer, 2001)]

Nachdem die Tabellenrechte der Rollen definiert sind, müssen dem Benutzer die Angaben in den Tabellen zur flexiblen Erstellung eines Berichtes dargestellt werden. Anhand dieser Informationen können die Daten für einen eigenen Report zusammengestellt werden. Die in der Datenbank enthaltenen Informationen werden ebenfalls über ein Schema abgebildet. Zu jeder Accounting-Tabelle oder -Datensicht werden *Metadaten* definiert. Metadaten bezeichnen allgemeine Daten, die Informationen über andere Daten enthalten. Sie werden nur intern vom Programm genutzt. Durch dieses Schema können verschiedene Datenquellen genutzt werden, in dem die Daten beschrieben werden. Es können daher jederzeit neue Datenbank-Tabellen hinzugefügt werden. Bei den beschriebenen Daten handelt es sich hier um die einzelnen Tabellen mit ihren Spaltennamen. Dem Benutzer werden verständliche Bezeichnungen für die Auswahl der Daten angeboten. Zu jedem Spaltennamen und seiner Bezeichnung werden weitere Eigenschaften gespeichert, die für das Anzeigen, die Gruppierung und zu treffende Einschränkungen wichtig sind. Diese Tabellendefinitionen werden ebenfalls in einer XML-Datei erfasst. In der Abbildung 5-10 ist die XML-Beschreibung der Tabelle *sc_jobs_j* dargestellt, die bereits in der Abbildung 5-8 gezeigt wurde.

```
<tables>
 <system name="softcomp">
  <db_table name="sc_jobs_j" description="job information from torque">
   <grouping name="user">
    <column name="username" description="user name" show="1" numeric="0" />
    <column name="userid" description="user account" show="1" numeric="0"
     category="user" />
    <column name="org" description="organisation" show="1" numeric="0"
     category="org" />
    <column name="groupid" description="group account" show="1" numeric="0" />
   </grouping>
   <grouping name="dates">
    <column name="subdate" description="entry date" show="1" numeric="0"
     category="date" />
    <column name="startdate" description="start date" show="1" numeric="0"
     category="date" />
    <column name="enddate" description="end date" show="1" numeric="0"
     category="date" />
   </grouping>
   <grouping name="job data">
```

```
   <column name="jobname" description="job name" show="1" numeric="0" />
   <column name="jobid" description="job number" show="1" numeric="0" />
   <column name="queue" description="queue" show="1" numeric="0" />
   <column name="cpus" description="number of cpu" show="1" numeric="1" />
   <column name="errval" description="error value" show="1" numeric="1" />
  </grouping>
  <grouping name="times">
   <column name="cputime" description="cpu time in sec" show="1" numeric="1"
    category="time" />
   <column name="walltime" description="wall time in sec" show="1" numeric="1"
    category="time" />
  </grouping>
  <grouping name="memory">
   <column name="mem" description="memory" show="1" numeric="1"
    category="memory" />
   <column name="vmem" description="virtual memory" show="1" numeric="1"
    category="memory" />
  </grouping>
  <grouping name="accounting">
   <column name="accno" description="accounting number" show="1" numeric="0" />
   <column name="contpool" description="contingent pool" show="1" numeric="0" />
   <column name="a_class" description="accounting class" show="0" numeric="0" />
   <column name="r_class" description="resource class" show="0" numeric="0" />
   <column name="ve" description="accounting units" show="1" numeric="1" />
  </grouping>
 </db_table>
</system>
</tables>
```

Abbildung 5-10 XML-Tabellendefinition vom Linux Cluster SoftComp

In der Datei werden die Tabellen zu jedem System aufgeführt. Die Zuordnung erfolgt durch das Element `<system name="softcomp">`, das in der nächst höher liegenden Ebene definiert wird. Zu einer Tabelle wird die sprechende Bezeichnung als Attribut hinzugefügt: `<db_table name="sc_jobs_j" description="job information from torque">`. Die einzelnen Spalten einer Tabelle werden in verschiedene Klassen eingeteilt, um die Übersichtlichkeit für den Benutzer zu verbessern. In der XML-Datei wird dazu das `<grouping>`-Tag genutzt. Die gesamte Struktur der Tabellen wird auf diese Weise gespeichert, um dem Benutzer alle Informationen anzubieten. Er kann dadurch selbst bestimmen, welche Daten aus der Datenbank selektiert werden sollen.

Auswahl der Informationen für einen Report – Konfiguration der Abfrage

Je nach Bedarf können die Informationen für einen Report individuell zusammengestellt werden. Der Inhalt und der Aufbau einen Berichtes sind somit abhängig von den Anforderungen des Erstellers.

Da die Schritte zur flexiblen Erstellung eines Reports immer in der gleichen Reihenfolge abgearbeitet werden, soll für den Benutzer eine einfache und nachvollziehbare Bedienung ermöglicht werden. Durch die Verwendung eines *Wizards* wird der Benutzer beim Verfassen seiner Berichte unterstützt. Ein Wizard bezeichnet eine Oberfläche, mit der ein Benutzer durch mehrere Auswahlmenüs geführt wird. Dadurch wird eine Hilfestellung gegeben. Der Wizard soll in einzelne Formulare untergliedert werden, die nacheinander folgen. Auf diese Weise lässt sich der Bericht intuitiv ohne zusätzliches Tutorial anfertigen. HTML-Formulare stellen eine

Schnittstelle zum Benutzer dar und dienen der Interaktion. Sie werden für die Eingabe und Auswahl der Daten verwendet. Hierbei können Eingabefelder ausgefüllt oder aus Listen Einträge ausgewählt werden. Formulare bestehen aus Elementen wie Texteingabefelder, Checkboxen, Radiogroups, Aufklapp-Menüs und Buttons. Der Vorteil von Formularen besteht darin, dass sie unabhängig vom Betriebssystem sind und dadurch von beliebigen Clients benutzt werden können. Nach dem Ausfüllen werden die Daten an den Webserver geschickt, um sie zu verarbeiten.

Für die Konfiguration der SQL-Abfrage soll eine Auswahlliste der angebotenen Informationen präsentiert werden. Diese Liste ist ein standardisiertes Formular und für jede Tabelle gleich aufgebaut. Die Spalten einer Tabelle werden dem Benutzer als auszuwählender Inhalt für den Bericht in der Reporting-Spalte angeboten. Außerdem kann er entscheiden, wie die Daten aufbereitet werden sollen z.B. nach welchen Kriterien sie verdichtet, sortiert oder gefiltert werden. Das tabellenartige Formular bietet dazu eine leichte und intuitive Bedienbarkeit, da die gewünschten Informationen nur durch Anklicken individuell ausgewählt werden sollen. In der Abbildung 5-11 wird der Entwurf zu diesem Auswahlformular dargestellt.

Reporting Information	**Gruppierung**					**Sortierung**		**Einschränkung**	
Spaltenbezeichnung	*no.*	*max*	*min*	*sum*	*avg*	*asc*	*desc*	*<, >, =*	*Wert*
⋮	⋮					⋮		⋮	

Abbildung 5-11 Entwurf des Auswahlformulars

Dem Benutzer soll die Möglichkeit gegeben werden, die gewünschten Daten gruppieren zu können. Er kann dabei folgende Auswahl treffen:

- Summierung der Informationen (sum)
- Zählen der Informationen (count)
- Finden von Maximum bzw. Minimum (min, max)
- Berechnung des Mittelwertes (avg)

Weiterhin sollen die Daten aufsteigend und absteigend sortiert werden können. Das Suchergebnis kann ebenso durch verschiedene Bedingungen eingeschränkt werden. Dazu werden mathematische Vergleichsoperatoren (=, >, <, ≠) benutzt. Um bestimmte Bereiche einzugrenzen zu können z.B. für Zeiträume, werden zusätzliche Operatoren verwendet. Durch dieses Formular entwickelt der Benutzer eigenständig eine SQL-Abfrage, ohne dass ihm die Syntax bekannt sein muss. Er kann dabei individuelle Informationen nach seinen eigenen Wünschen zusammenstellen. Die flexible Erstellung soll in der Regel für die kurzfristigen Abweichungs- und Bedarfsberichte genutzt werden.

Abruf der Daten und Generieren des Reports.

Hat der Benutzer eine Auswahl getroffen, werden jeweils die aktuellen Daten aus der Datenbank ausgegeben. Auf diese Daten kann schnell zugegriffen werden, in dem eine SQL-Abfrage zu den ausgewählten Informationen generiert wird. Die Berichte werden somit immer dynamisch zur Laufzeit erstellt. Die Ausgabe der Daten und somit die Darstellung der Reports auf der Webseite werden im nächsten Unterkapitel Reportdarstellung beschrieben.

5.3.3. Standard- und benutzereigene Reports

Die Standard-Reports werden neben der flexiblen Erstellung zur Verfügung gestellt und sollen zunächst den allgemeinen Bedarf an Informationen abdecken. Sie sind vom Administrator erstellte Vorlagen, die einheitlich definiert und für jeden Benutzer zugänglich sind. Für diese Berichte ist bereits eine Auswahl an Informationen festgelegt, die auch individuell angepasst werden können. Somit können die Standard-Reports als Ausgangsbasis für flexible Berichte dienen. Die Vorlage kann dabei nachträglich abgewandelt werden, in dem über das Auswahlformular Änderungen vorgenommen werden. Für eine benutzereigene Vorlage können die eigens ausgewählten Informationen zu einem Bericht gespeichert werden. Wenn das Reporting-System zu einem späteren Zeitpunkt benutzt wird, kann diese Vorlage wieder verwendet werden. Dabei muss der Zeitbezug der gewünschten Daten für spätere Abfragen beachtet werden, damit sie sich nicht auf bereits abgelaufene Perioden beziehen. Das Reporting-System soll dabei nicht unterscheiden, ob es sich bei den Berichten, um einen Standard- oder einen benutzereigenen Bericht handelt. Das Verfahren soll immer gleich sein. Bei diesen Vorlagen wird die komplette Auswahl der gewünschten Daten abgelegt, jedoch nicht die Daten selbst. Die ausgewählten Informationen sollen als strukturiertes XML-Schema abgelegt werden. In der Abbildung 5-12 wird die XML-Definition eines benutzereigenen Reports dargestellt. In diesem Fall hat ein Benutzer der Rolle *user* die Felder *enddate, walltime, cputime* der detailierten Tabelle *sc_jobs_j* für das Linux Cluster SoftComp ausgewählt. Der daraus zu erstellende Bericht ermittelt jeweils den Mittelwert und die Summe für die *walltime* und *cputime* gruppiert nach dem Feld *enddate*. Die Daten werden dabei nach *enddate* aufsteigend sortiert ausgegeben. Weiterhin sind die einzuschränkenden Bedingungen zu den einzelnen Datenfeldern gespeichert, die auch die ausgewählten benutzereigenen Projekte und BenutzerIDs enthalten.

```
<report>
        <role>user</role>
        <system>softcomp</system>
        <level>detailed</level>
        <table>sc_jobs_j</table>
        <select>enddate</select>
        <select>walltime</select>
        <select>cputime</select>
        <avg>walltime</avg>
        <avg>cputime</avg>
        <sum>walltime</sum>
        <sum>cputime</sum>
        <asc>enddate</asc>
        <condition name="cputime">
                <gt>0</gt>
        </condition>
        <condition name="enddate">
                <BETWEEN>01-JAN-09</BETWEEN>
                <BETWEEN>01-APR-09</BETWEEN>
        </condition>
        <org>projectid1</org>
        <user>userid1</user>
        <user>userid2</user>
        <user>userid3</user>
</report>
```

Abbildung 5-12 XML-Definition eines benutzereigenen Reports

Der Aufbau des Dokumentes ist dabei vorgegeben. Dadurch unterscheiden sich auch die Standardberichte nicht wesentlich von den Benutzereigenen. Die benutzereigenen Berichte werden nur durch die Rechte des einzelnen Benutzers eingeschränkt und unterscheiden sich dadurch von den Standardisierten. In diesen Berichten werden dazu benutzerdefinierte Daten wie z.B. die Rolle gespeichert.

Um einen Standard- oder benutzereigenen Report zu erzeugen, müssen die drei Schritte zur flexiblen Erstellung (siehe Abschnitt 5.3.2) nicht einzeln umgesetzt werden. So sind das Bereitstellen und die Auswahl der Informationen schon durch das XML-Schema vorgegeben. Lediglich die Daten müssen aus der Datenbank abgerufen und der Report generiert werden. Der Bericht wird somit immer aktualisiert.

Für die Standardberichte soll ein fester Katalog bereitgestellt werden. Dieser beinhaltet dann sowohl vorhandene als auch neu zu entwickelnde Berichte. Außerdem ist es jederzeit möglich, den Katalog zu erweitern oder zu vervollständigen. Als Beispiele für den Katalog der Standardberichte können die bereits existierenden Reports im JSC wie z.B. die Kontingentinformationen zu den Projekten (Abschnitt 3.2.4) benutzt werden.

5.4. Reportdarstellung

Für die Berichte soll die Präsentation der ausgewählten Informationen in optisch aufbereiteter Form erfolgen. Dafür gibt es zwei verschiedene Arten der Darstellung. Einerseits sollen die Daten in Tabellen oder Listen dargestellt. Andererseits können die Daten auch in Diagrammen abgebildet werden. In den folgenden Abschnitten wird auf die unterschiedlichen Darstellungsweisen genauer eingegangen.

5.4.1. Darstellung der Daten

Wie bereits in Abschnitt 5.2.2 beschrieben, wird für das Accounting ein relationales Datenbankmodell verwendet. In diesem Modell wird eine Menge von Daten in einer Relation gespeichert. Die Relation wird als zweidimensionale Tabelle dargestellt und in Zeilen und Spalten strukturiert. Daher wird ein Ausschnitt der Datenbankrelation in den vom Reporting-System ausgegebenen Daten widergespiegelt.

Um die Daten in einer strukturierten Form anzuzeigen, sollen sie in einer Tabelle oder Liste auf der Webseite des Reporting-Systems dargestellt werden. Diese Darstellungsformen dienen der Übersichtlichkeit und der Verständlichkeit, da die Daten in einer geordneten Zusammenstellung wiedergegeben werden. Der darzustellende Inhalt in einer Tabelle ist in Zeilen und Spalten gegliedert und grafisch ausgerichtet. In der ersten Zeile (Kopfzeile) werden die Namen der zuvor ausgewählten Daten als Spaltennamen angezeigt. In den nachfolgenden Zeilen werden jeweils die miteinander verknüpften Datensätze ausgegeben. Alle Werte, die zu einer Datenreihe gehören, stehen in einer Spalte. Es besteht also ein semantischer Zusammenhang zwischen dem Inhalt eines einzelnen Datenfeldes (Zelle) und der Spalte bzw. der Zeile, in der er sich befindet. Der Sonderfall einer eindimensionalen Tabelle ist eine Liste. Dabei wird der Inhalt der Liste als eine Spalte mit der dazugehörigen Bezeichnung ausgegeben. Die Liste ermöglicht ebenfalls eine überblickartige Orientierung und Vergleichsmöglichkeiten.

Die Daten in der Tabelle oder Liste sind bereits sortiert oder gruppiert, wenn es der Benutzer beim Erstellen eines Berichtes entsprechend festlegt. Dabei können die Daten aufsteigend oder absteigend sortiert sein. Je nach Art der Gruppierung werden die gewünschten Informationen zusammengefasst.

Dem Benutzer soll die Möglichkeit gegeben werden, die erstellten Reports zu speichern. Dadurch können sie später auch offline verwendet werden. Die Dateien werden dazu vom Webserver an den Client übertragen. Dabei muss darauf geachtet werden, dass die Berichte für jeden Benutzer speziell abgelegt werden. Die Ergebnisse der Datenbankabfrage sollen so gesichert werden, dass sie auch von anderen Programmen genutzt werden können. Die gebräuchlichsten Formate für tabellarisch dargestellte Daten sind *CSV* und *XLS*.

Das Dateiformat *CSV* (Comma Separated Value) beschreibt den Aufbau einer Textdatei zur Speicherung oder zum Austausch. Der Inhalt der CSV-Datei ist so formatiert, dass jede Zeile einen Datensatz enthält und die einzelnen Felder durch Kommata voneinander getrennt werden. Dadurch können einfach strukturierte Daten wie in Tabellen oder Listen variabler Länge abgebildet werden. Der erste Datensatz in dieser Datei entspricht der Kopfzeile der Tabelle, den Spaltennamen. Die Formatierung der Daten selbst wird beim Speichern in der CSV-Datei nicht verändert. Der Vorteil bei der Verwendung von CSV besteht darin, dass die Daten mit jeder beliebigen Programmiersprache leicht eingelesen werden können. Dieses Format ist außerdem sehr weit verbreitet. Es kann auch von anderen Anwendungen wie z.B. Tabellenkalkulationsprogrammen und Datenbanksystemen importiert und exportiert werden. In der Abbildung 5-13 wird ein Beispielreport als CVS-Datei angezeigt.

```
user account,wall time in sec
userid1,6117983
userid2,18694411
userid3,32833240
userid4,33787745
userid5,79533387
```

Abbildung 5-13 CSV-Datei eines Reports

XLS-Dateien (Excel Spreadsheet) basieren auf einem offenen Binärformat von Microsoft. Es wird vom Tabellenkalkulationsprogramm Microsoft Excel verwendet und erstellt. Microsoft Excel ist die meistverbreitete Software für Tabellenkalkulation. Der auf der Webseite dargestellte Report wird in der XLS-Datei in gleicher Weise angezeigt.

5.4.2. Grafische Darstellung

Die Daten eines Berichtes sollen zusätzlich durch eine grafische Darstellung abgebildet werden. Somit kann ein bestimmter Sachverhalt besser verdeutlicht werden. Im Vergleich zur textlichen Darstellung können Grafiken schneller erfasst werden, obwohl mit ihnen nicht jeder einzelne Aspekt beschrieben werden kann. Ein wesentlicher Vorteil ist die bessere Anschaulichkeit von grafischen Darstellungen. Außerdem können mit Hilfe der grafischen Darstellung die Zusammenhänge der einzelnen Daten deutlicher dargestellt werden.

Grafische Darstellungen können als Bilder in einer Vielzahl von Formaten gespeichert werden. Für die grafische Darstellung der Reports muss somit ein geeignetes Format gefunden werden. Die gebräuchlichsten Formate sind *GIF (Graphics Interchange Format)*, *JPEG (Joint Photographic Expert Group)* und auch *PNG (Portable Network Graphics)*. Mit diesen

Formaten werden Pixelgrafiken erzeugt, die aus einem Raster vieler kleiner *Pixel* (Bildpunkte) bestehen. Die Pixel einer Grafik können neben den *RGB*-Farbwerten *(Rot, Grün, Blau)* auch einen Transparenzwert enthalten. Dabei bilden die Bildpunkte der Grafik eine Art Mosaik, das aus der Ferne betrachtet eine natürliche Form ergibt. Aus der Nähe oder beim Vergrößern werden diese Pixel sichtbar. Je genauer ein Bild bzw. je besser die Bildauflösung sein soll, desto mehr Pixelinformationen müssen abgespeichert werden. Bei der Anzeige von Webseiten ist jedoch die Größe der Datei für die Übertragung entscheidend, da der Benutzer sonst relativ lange Wartezeiten hinnehmen muss. Bei einer kleinen Dateigröße gehen aber die Detailinformationen einer Grafik verloren, so dass der Report unscharf und ungenau dargestellt wird.

Im Gegensatz zu Pixelgrafiken beschreiben vektorgrafische Formate die Bildinformationen über Vektoren. Es werden dadurch nur die Beschreibungen der Objekte mit ihrer Position und Ausmaßen, Farben und andere Eigenschaften in der Datei gespeichert. Beispielsweise werden für die Darstellung eines Kreises mindestens zwei Werte benötigt: die Lage des Mittelpunktes und der Kreisradius. Die anzuzeigende Grafik wird dann zur Laufzeit aus den beschreibenden Daten erzeugt. Sie eignet sich ebenfalls zur Anzeige im Web, da viel weniger Speicherplatz benötigt wird als bei pixelorientierten Grafiken. Ein weiterer Vorteil von Vektorgrafiken ist, dass sie sich verlustfrei zoomen lassen (Abbildung 5-14). So können kleine Grafiken beliebig vergrößert werden und die Qualität geht dabei nicht verloren. Vektorgrafiken können auch in Pixelgrafiken umgewandelt werden (*Rasterung*), umgekehrt ist dies jedoch wesentlich schwieriger und fehlerbehaftet.

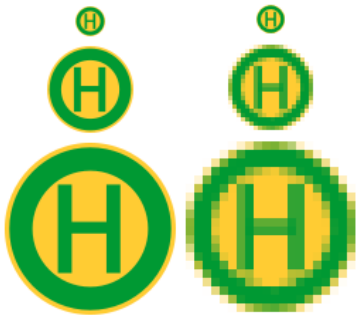

Abbildung 5-14 Vektor- versus Pixelgrafik[7]

Ein offizieller Standard für Vektorgrafiken ist *SVG (Scalable Vector Graphics)*. Er dient zur textuellen Beschreibung von 2D-Grafiken, Texten und Rasterbildern und unterstützt Animationen und Interaktivität. SVG wird in der Technischen Illustration und Dokumentation, sowie in der Kartografie und für die Visualisierung von Daten eingesetzt. SVG basiert auf XML, die Datei besteht also aus Markup Language Tags, mit denen Linien, Rechtecke, Kreise, Polygone, Ellipsen und Text definiert werden können. Jede Grafik wird durch diese einfachen Elemente zusammengesetzt. Außerdem lassen sich auch andere Grafiken oder Objekte in SVG einbinden. Durch XML ist eine strukturierte Ablage der einzelnen Daten möglich, d.h. auf einzelne Elemente und Eigenschaften einer Grafik können gezielt zugegriffen werden. Der Vorteil ist dabei die semantische Struktur.

Eine Alternative wäre *Macromedia Flash*, dies ist jedoch kein offener Standard wie SVG. Außerdem handelt es sich bei Flash um ein binäres Datenformat. Bei dem Vergleich der

[7] Von http://de.wikipedia.org/wiki/Vektorgrafik

Erstellung zeigt sich, dass XML-basierte Formate wesentlich leichter integriert werden können als binäre Dateien. Durch die Beschreibung in Klartext können SVG-Grafiken z.B. mit Skriptsprachen erzeugt und auch jederzeit mit Hilfe eines Texteditors geändert werden. Für SVG wird daher kein bestimmtes Tool benötigt. Bei Flash wiederum ist die Erstellung von dem eigenen Editor und der eigenen Skriptsprache abhängig.

SVG wurde in erster Linie für den Einsatz im Web konzipiert. Insbesondere bei dynamischen Webseiten können die grafischen Darstellungen mit SVG umgesetzt werden. Dies kann vor allem für die serverseitige Generierung genutzt werden. Die darzustellenden Daten können bei der dynamischen Erzeugung auch aus Datenbanken stammen. Um den Inhalt der SVG-Grafik anzuzeigen, wird ein Browser benötigt, der XML verarbeiten kann und die Grafik rendert. Für die Browser des Mozilla-Projektes wurde ein nativer SVG-Viewer entwickelt. Viele andere Browser außer dem Internet Explorer können ebenfalls SVG ohne ein zusätzliches SVG-Programm darstellen. Von Adobe wird außerdem ein SVG-Plug-In[8] angeboten, das von allen Browsern und Betriebssystemen unterstützt wird. Ein Plug-In wird ebenfalls für die Anzeige von Flash-Grafiken benötigt.

SVG ist also derzeit der elegantere Ansatz für vektorielle Grafiken im Web. Es ist plattformübergreifend, frei verwendbar und als Grafikaustauschformat entworfen. SVG erfährt außerdem eine kontinuierliche Weiterentwicklung und Erweiterung seiner Funktionalität. Für das Reporting-System wird wegen der beschriebenen Vorzüge sowohl zur Pixelgrafik als auch zu Macromedia Flash das Format SVG verwendet. (Neumann & Winter, 2001)

Durch die Erstellung der Grafiken im SVG-Format kann das Gsharp-Programm (siehe Abschnitt 3.2.1) ersetzt werden. Durch die skalierbare Auflösung können ebenfalls hochqualitative Bilder erzeugt werden. Damit der Benutzer die erstellte Grafik verwenden kann, wird die SVG-Datei zum Herunterladen angeboten. Sie können dann in unterschiedlichen Präsentationsmedien z.B. auf Papier in Geschäftsberichten oder auf Postern genutzt werden.

Als grafische Darstellungen von Daten oder Informationen werden Diagramme verwendet. Für die Veranschaulichung der Zusammenhänge zwischen abhängigen Werten werden Achsendiagramme benutzt. Dabei wird die Abbildung von Größen- und Mengenverhältnissen anschaulicher und es sind einfache Vergleiche möglich. In den Diagrammen werden die zuvor aufgelisteten Daten aufbereitet dargeboten. Daraus können nun leichter Schlussfolgerungen gezogen werden, die der Benutzer bzgl. seiner Anfrage an das Reporting System gestellt hat. Zur Analyse der Daten werden unterschiedliche Diagramme bereitgestellt. Dabei können jeweils verschiedene Aussagen und Verwendungszwecke des Erstellers abgebildet werden. Der Inhalt der Diagramme ist von den Wünschen des Benutzers abhängig, da er sich die Daten zunächst selbst auswählt. Falls der Benutzer sortierte Daten ausgesucht hat, werden sie auch in der Grafik sortiert dargestellt. Aus der praktischen Erfahrung des bestehenden Reporting beschränkt sich die Auswahl hier auf drei verschiedene Diagrammarten: das Säulen-, das Linien- und das Kreisdiagramm. Im Folgenden werden die verwendeten Diagrammformen genauer betrachtet.

In dem Säulendiagramm veranschaulichen senkrecht auf der x-Achse stehende Rechtecke die Datenpunkte. Dabei geben die abgebildeten Säulenlängen das Verhältnis an. Im Hintergrund werden dafür Gitternetzlinien angezeigt. Zusätzlich werden die Säulen mit den dazugehörigen

[8] http://www.adobe.com/svg/viewer/install/main.html

Datenwerten beschriftet. Ein Säulendiagramm eignet sich nicht für alle Daten, da eine Säule optisch alle Werte vom Ursprung bis zum Endwert einschließt. Weiterhin ist darauf zu achten, dass nicht zu viele Datenreihen und Werte die Klarheit und Übersichtlichkeit beeinträchtigen. Die Anzahl der Säulen soll daher überschaubar sein. Bei vielen Kategorien leidet die Anschaulichkeit, daher sollte ein Liniendiagramm bevorzugt werden. Falls mehrere Datenreihen zu einer Bezugsgröße ausgewählt werden, wird ein überlappendes Säulendiagramm dargestellt. Der Schwerpunkt wird somit auf die einzelnen Datenmengen und nicht auf deren Summe gelegt, so dass zunächst keine gestapelten Säulen verwendet werden. Die Säulen mehrerer Reihen werden farblich verschieden abgebildet. Durch die Verwendung von SVG kann die Transparenz der Säulen bei der überlappenden Anzeige eingestellt werden. Die Säulendiagramme dienen zum Vergleich von absoluten Werten miteinander. Weiterhin eignen sie sich, um den zeitlichen Verlauf der veränderten Daten wiederzugeben und deren Entwicklung aufzuzeigen. Durch das mehrmalige Auftreten eines bestimmten Objektes in einer Reihe, wird ein Häufigkeitsvergleich ermöglicht. In der Abbildung 5-15 wird der Verbrauch einzelner Benutzer eines Projektes angezeigt. Eine solche Grafik kann z.B. vom Projektleiter erstellt werden.

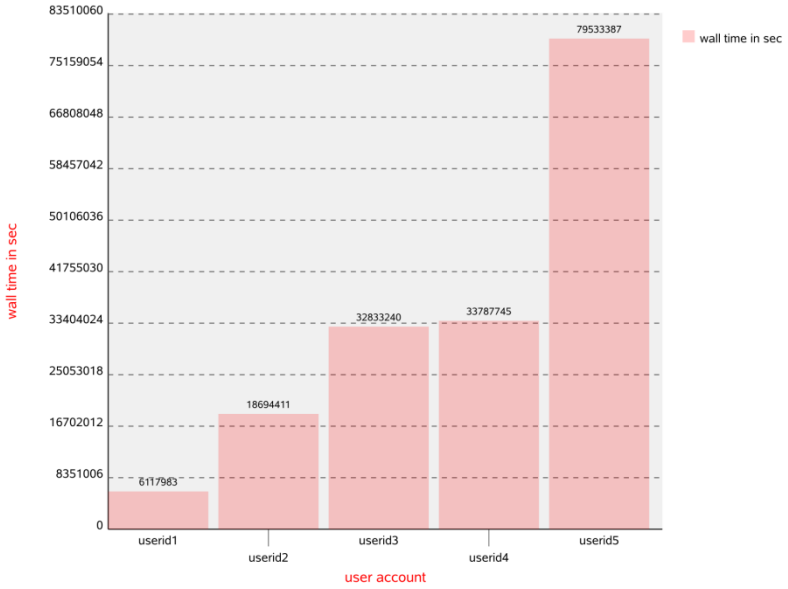

Abbildung 5-15 Darstellung eines Reports als Balkendiagramm

In einem Liniendiagramm werden die Datenpunkte miteinander durch Linien verbunden. Dadurch wird in der grafischen Darstellung ein funktionaler Zusammenhang zweier oder mehrerer Merkmale erkennbar wie z.B. bei einer mathematischen Formel. Mit einem Liniendiagramm wird ein anschaulicher Verlauf der Daten erreicht und deren Entwicklung abgebildet. Bei einfachen Liniendiagrammen lassen sich alle möglichen Zahlenbereiche darstellen. Mehrere Datenreihen werden durch die unterschiedlichen Linienfarben voneinander abgehoben. Ungeeignet ist die Darstellung, wenn sich die Linien überlagern. Auch wenn mehrere Datenpunkte angegeben werden, bleibt die Übersichtlichkeit erhalten. Sie sind also besonders geeignet für Daten, die über längere Zeiträume hinweg erhoben werden. Mit Liniendiagrammen lassen sich somit auch Trends im Rahmen einer kontinuierlichen Weiterentwicklung ableiten. In der Abbildung 5-16 werden die Laufzeiten verschiedener Jobs eines Benutzers dargestellt.

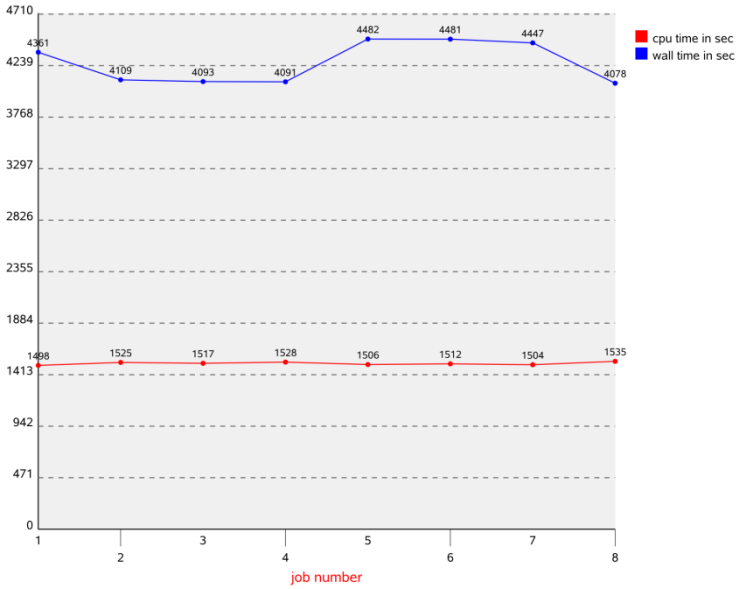

Abbildung 5-16 Darstellung eines Reports als Liniendiagramm

Ein Kreisdiagramm wird wahlweise auch als Kuchen- oder Tortendiagramm bezeichnet. Es ist eine Darstellungsform für Einzelwerte als Teil eines Ganzen. Die Werte werden in Form von Kreissektoren abgebildet. Das charakteristische Merkmal ist dabei der Winkel, der die Größe des Sektors bestimmt. Dadurch werden die Größenverhältnisse ihrer Anteile und auch die Zusammensetzung des Ganzen ersichtlich. Die Verwendung von Kreisdiagrammen ist also sinnvoll, wenn die dargestellten Daten eine Gesamtmenge ergeben. Bei Teilmengen sind Säulendiagramme zu bevorzugen. Diese Diagramme eignen sich besonders für die Darstellung von Verteilungen und Anteilen. Dabei können sich die Daten auf einen bestimmten Zeitpunkt beziehen. Die unterschiedlichen Sektoren werden beim Erstellen der Grafik nicht nach der Größe sortiert, da der Benutzer schon beim Erstellen des Reports angibt, ob die Daten sortiert werden sollen. Zur besseren Unterscheidung der jeweiligen Sektoren werden verschiedene Farben verwendet. Die Beschriftung erfolgt am Rand des Kreissektors. Dabei wird der Prozentsatz des Anteils angegeben. Der gesamte Kreis entspricht also hundert Prozent. Bei Kreisdiagrammen ist nur die Darstellung jeweils einer Datenreihe möglich. Dabei sollen auch nicht zu viele Datenwerte angegeben werden, da ansonsten die Übersichtlichkeit verloren geht. In der Legende werden den Sektoren die Beschreibung und der eigentliche Datenwert zugeordnet. Das Diagramm wird in einer 2D-Darstellung angezeigt, da mit dem 3D-Effekt der Kreis zu einer Ellipse gestaucht wird und eine verzerrende Wirkung auftritt. Mit Hilfe der folgenden Abbildung 5-17 wird ein Bericht der Monatsauswertungen gezeigt, der vom Administrator erstellt werden kann. In diesem Report wird die Auslastung der einzelnen Benutzergruppen eines Rechnersystems dargestellt, wie auch schon in Abschnitt 3.2.1 in Abbildung 3-4 gezeigt.

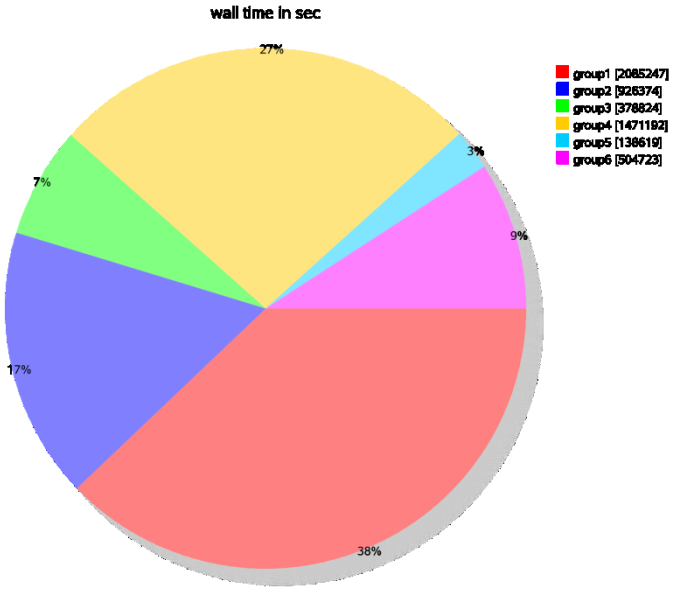

Abbildung 5-17 Darstellung eines Reports als Kreisdiagramm

Kapitel 6 Realisierung

In diesem Kapitel wird die Implementierung des in Kapitel 5 entwickelten Konzeptes beschrieben. Dabei wird zunächst auf die Konfiguration des Webservers eingegangen, um die HTTPS-Kommunikation und die Benutzerauthentifizierung über Zertifikate zu realisieren. Im Anschluss wird die Wahl der Programmiersprache begründet. Danach wird die Programmstruktur mit den einzelnen Modulen dargestellt und erläutert. Abschließend wird der Programmablauf veranschaulicht, um den Mechanismus der Webanwendung nachvollziehen zu können.

6.1. Webserver

Der Webserver wird auf einem Linux-System mit der Open-Source-Software Apache eingerichtet. Den Kern von Apache bildet die vollständige Implementierung des HTTP-Standards. Daneben gibt es viele Erweiterungen für Spezialaufgaben, diese werden in Form von Modulen realisiert. Die Einstellungen des Apache-Webservers werden über Direktiven in den Konfigurationsdateien vorgenommen.

Für die Webanwendung muss die CGI-Schnittstelle unterstützt werden, die das Modul `mod_cgi` anbietet. Das entsprechende CGI-Programm wird dazu als externer Prozess gestartet. Es enthält die Formulardaten als Eingabe und leitet die Ausgabe an den Client weiter. Bei der Standardinstallation von Apache ist das Modul bereits aktiv.

Für die Verwendung von SSL muss zunächst ein Schlüsselpaar für das Server-Zertifikat generiert werden. Dieses Zertifikat wird von einer vertrauenswürdigen Organisation signiert. Durch das Zertifikat werden dann der Webserver und die Domain eindeutig identifiziert. Um die abgesicherte HTTPS-Kommunikation zu nutzen, muss der Apache-Webserver konfiguriert werden. Alle zwischen dem Client und dem Server ausgetauschten Daten werden dadurch verschlüsselt und digital signiert. Dazu muss das Modul `mod_ssl` geladen werden. Die Kommunikation läuft über den Port 443, für den die Funktionalität über SSL-Direktiven angepasst wird.

Im Apache-Webserver lässt sich die Benutzerauthentifizierung über Zertifikate ebenfalls durch die Verwendung des Moduls `mod_ssl` realisieren. Dabei werden der Aussteller, die Unterschrift des Austellers und der Gültigkeitszeitraum automatisch überprüft. Die Authentisierung erfolgt dabei im Rahmen des SSL-Protokolls. Sobald ein Client auf eine bestimmte Ressource zugreifen will, benötigt er daher ein gültiges SSL-Zertifikat. Der Zugriff auf das CGI-Skript ist somit nur mittels Client-Authentifizierung zugelassen. Dazu werden die zusätzlichen Konfigurationen ebenfalls in einer Datei gespeichert. Durch die Verwendung des Modules können zusätzliche Umgebungsvariablen definiert werden. Diese Variablen beziehen sich auf die Zertifikate und die HTTPS-Verbindung, die dann im CGI-Programm zur Verfügung stehen.

Bei dieser Konfiguration muss keine Benutzerverwaltung auf dem Webserver durchgeführt werden. Die Gewährung und Sperrung des Zugriffs geschieht durch das Ausstellen und Sperren von Zertifikaten. (Kersken, 2006)

6.2. Wahl der Programmiersprache

In Abschnitt 3.4.2 Randbedingungen und Vorgaben wurde festgestellt, dass das Reporting-System auf das Accounting-System aufbauen soll. Dieses System wurde bereits in der Programmiersprache Perl implementiert. Außerdem wurde für die Top-User-Listen (Abschnitt 3.2.2) ein Perl-Programm entwickelt. Wenn diese Sprache ebenfalls für das Reporting-System genutzt wird, können diese existierenden Reports eingebunden werden. Außerdem wird das Accounting-System und das neu zu erstellende Reporting-System gemeinsam von einem Administrator verwaltet und gepflegt. Für die Schnittstelle zum Client soll die Webanwendung mit CGI realisiert werden. Perl ist für die CGI-Programmierung die am weitesten verbreitete und meist benutzte Sprache und schon auf fast allen Webservern installiert. Weiterhin wird das effiziente Abspeichern von Daten aus den Datenbanktabellen durch bereits existierende Datenstrukturen wie z.B. einen Hash ermöglicht. Im *CPAN (Comprehensive Perl Archive Network)* werden zahlreiche frei verfügbare Bibliotheken zur Verfügung gestellt. So gibt es Perl-Module für die Schnittstelle zur Datenbank, ebenso wie die Verarbeitung von XML und die Erzeugung dynamischer Grafiken u.a. SVG. Der Einsatz der plattformabhängigen Programmiersprache Perl ist daher sinnvoll. (Guelich, Gundavaram, & Birznieks, 2001)

6.3. Programmstruktur

Für die Software-Entwicklung wird das Model-View-Controller-Konzept (MVC) verwendet. Es gliedert sich in drei Ebenen: das Modell, die Präsentation und die Steuerung. Ziel des Konzeptes ist es, einen flexiblen Programmentwurf zu entwickeln, der eine spätere Änderung oder Erweiterung erleichtert und die Wiederverwendbarkeit der einzelnen Komponenten ermöglicht. Das MVC-Konzept wurde speziell für grafische Benutzeroberflächen entwickelt. Die Komponenten hängen dabei unterschiedlich stark voneinander ab. Bei dieser Webanwendung kümmert sich der serverseitige Controller um die Interaktion mit dem Benutzer und wertet die eintreffenden Daten (HTTP-Requests) des Browsers aus. Weiterhin tritt er als aktiver Vermittler zwischen Model und View auf. Der Controller ist im Reporting-System das Hauptprogramm, also die Logik der Anwendung. Die Präsentationsebene wird durch das CGI-Ausgabe-Modul dargestellt, das den HTML-Code für den HTTP-Response erzeugt und somit das Aussehen der Webanwendung festlegt. Das Modell liefert die Daten und wird durch verschiedene Module realisiert. Im Einzelnen handelt es sich um das Datenbank-, das Login-, das XML- und das Grafik-Modul. In der Abbildung 6-1 wird der Zusammenhang der einzelnen Module dargestellt. Durch diese Dreiteilung werden die Daten von der visuellen Repräsentation getrennt. Die grafischen Komponenten können weiterentwickelt werden ohne das Model zu ändern.

Durch die Module wird eine Kapselung der Programmteile vorgenommen. Die Funktionen der unterschiedlichen Module werden vom Hauptprogramm oder auch von Funktionen anderer Module genutzt. Die Programmlogik kann dabei erneut genutzt werden, ohne dass der Programmcode redundant erstellt und gepflegt werden muss. Das Datenbank-, das Grafik- und das CGI-Ausgabe-Modul sind universell programmiert, so dass sie auch von anderen Programmen wiederverwendet werden können. Da das Reporting-System ein komplexes Programm ist, wird durch den zweckgebundenen Einsatz der Module und Funktionen der Programmcode gegliedert und strukturiert. (Ullenboom, 2009)

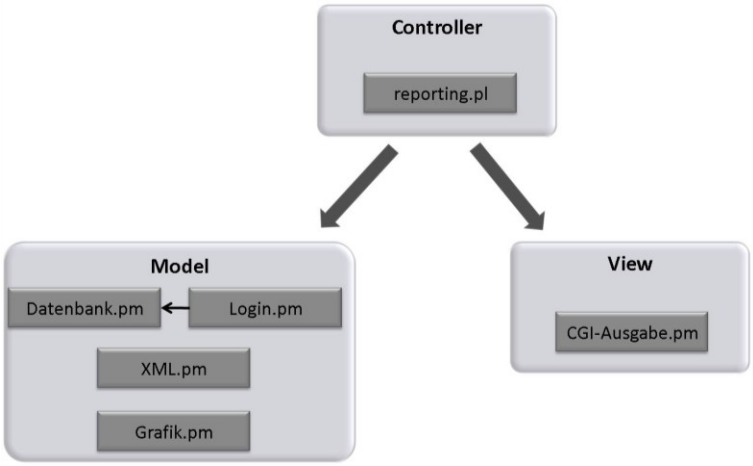

Abbildung 6-1 Programmstruktur des Reporting-Systems

In der folgenden Aufzählung wird die Funktionsweise der Reporting-System-Module genauer erläutert:

- Das *CGI-Ausgabe-Modul* verwendet das `CGI`-Modul, ein Standardwerkzeug zum Schreiben von CGI-Skripten. Es bietet relativ einfache Funktionen, um auf die Informationen zuzugreifen, die vom Server an das Programm gesendet werden, und um die CGI-Antwort aufzubereiten, die der Server erwartet. Außerdem besitzt das Modul Hilfsfunktionen für das Erzeugen von HTML – von Tabellen bis hin zu Formularwidgets. Zudem können die Daten aus den Formularen eingelesen und verarbeitet werden. Das Modul ist bereits in der Standarddistribution von Perl enthalten. Das CGI-Ausgabe-Modul stellt somit für das Reporting-System die Funktionen bereit, die alles was auf der Webseite zu sehen ist insbesondere die verschiedenen Formulare ausgeben.
- In dem *Datenbank-Modul* werden verschiedene Funktionen definiert, um die SQL-Abfragen zu generieren und die Daten aus der Datenbank zu lesen. Dafür werden in dem Modul die Perl-Module `DBI` und `DBIx::Dump` verwendet. `DBI` (DataBase Interface) bietet eine Datenbank-Schnittstelle für die Programmiersprache Perl. Sie unterstützt die wichtigsten Datenbankensysteme, so auch Oracle. Mit den Funktionen und Variablen des Moduls kann auf die Datenbank zugegriffen werden. Mit dem CPAN-Modul `DBIx::Dump` können die Ergebnisse einer relationalen Datenbank-Abfrage in Dateien abgelegt werden. Dabei werden die Excel- und CSV-Dateiformate unterstützt, die von anderen Programmen genutzt werden können. Für den Verbindungsaufbau zur Datenbank werden der Benutzername und das Passwort in separaten Hilfsdateien abgespeichert.
- Das *Login-Modul* enthält die Funktionen zur Zugangs- und Zugriffskontrolle. In diesen Funktionen werden die Abfragen zur Identifikation eines Reporting-System-Benutzers sowie auch für die verschiedenen Rollen definiert.
- Mit dem *Grafik-Modul* kann ein Achsendiagramm im SVG-Format erstellt werden. Insbesondere werden Funktionen zur Erzeugung von Säulen-, Linien- und Kreisdiagrammen angeboten. Die CPAN-Module von `SVG::TT::Graph` bieten eine einfache Möglichkeit zur Erstellung von SVG-Diagrammen. Durch die verschiedenen Optionen werden die Graphen konfiguriert und das Design vorgegeben. Um die

Diagramme im Web anzuzeigen, testet eine Funktion, ob der Browser SVG darstellen kann.

- Das *XML-Modul* bietet Funktionen zum Einlesen und Schreiben von XML-Dateien. Die Struktur und der Inhalt einer solchen Datei werden in einer entsprechenden Datenstruktur abgelegt. Ebenso kann aus einer Datenstruktur eine XML-Datei erzeugt werden. Das verwendete Modul `XML::Simple` aus dem CPAN liest die Daten mit einen speziellen XML-Reader (Parser) ein. Mit Hilfe der erzeugten Datenstruktur wird der beliebige Zugriff auf alle Elemente ermöglicht.
- Das *Fehler-Modul* wird verwendet, um Fehlermeldungen auszugeben. Es wird dabei ein eigener Handler zur Fehlerbehandlung benutzt, auf den alle Module zugreifen können.

Auf den Ablauf des Hauptprogramms wird im folgenden Unterkapitel genauer eingegangen.

6.4. Programmablauf

Zum Starten des Programms muss vom Benutzer die URL der Webanwendung im Browser eingegeben werden. Dabei wird die ausführbare Datei, das Hauptprogramm, aufgerufen. Der Programmablauf wird zunächst in der Abbildung 6-2 veranschaulicht, um einen ersten Überblick zu erhalten. In den folgenden Abschnitten wird dann der Inhalt dieser Darstellung erläutert. Dazu wird auf die einzelnen Teile der Konzeptionierung genauer eingegangen.

6.4.1. Zugangs- und Zugriffskontrolle

Durch die Konfiguration des Webservers (Abschnitt 6.1) wird bereits die Authentifizierung und damit der wesentliche Teil der Zugangskontrolle realisiert.

Durch den Abgleich mit der Datenbanktabelle `d_pers` wird das Zertifikat in der Datenbank gefunden, da es dem Benutzer eindeutig zugeordnet ist. Alle Benutzer, die kein gültiges oder angenommenes Zertifikat besitzen, werden auf eine Registrierungsseite umgeleitet. Ebenso werden Benutzer mit einem gültigen Zertifikat, das aber noch nicht in der Datenbank erfasst ist, auf diese Webseite geführt. Die Webanwendung ist an dieser Stelle beendet. Auf der Registrierungsseite befindet sich ein Hinweis zur Beantragung eines Zertifikates. Falls der Benutzer ein Zertifikat besitzt, wird der DN des Zertifikates ausgelesen und er muss seine Emailadresse angeben. Für den neuen Benutzer wird dann der Zertifikatschlüssel in der Tabelle zu seiner eindeutigen Emailadresse abgelegt. Dabei wird auch überprüft, ob der Benutzer zugelassen oder zurückgewiesen wird. Für neue Benutzer, die noch nicht in der Datenbank erfasst sind, kann bei der Beantragung eines Accounts oder eines Projektes schon ihr Zertifikat angegeben werden.

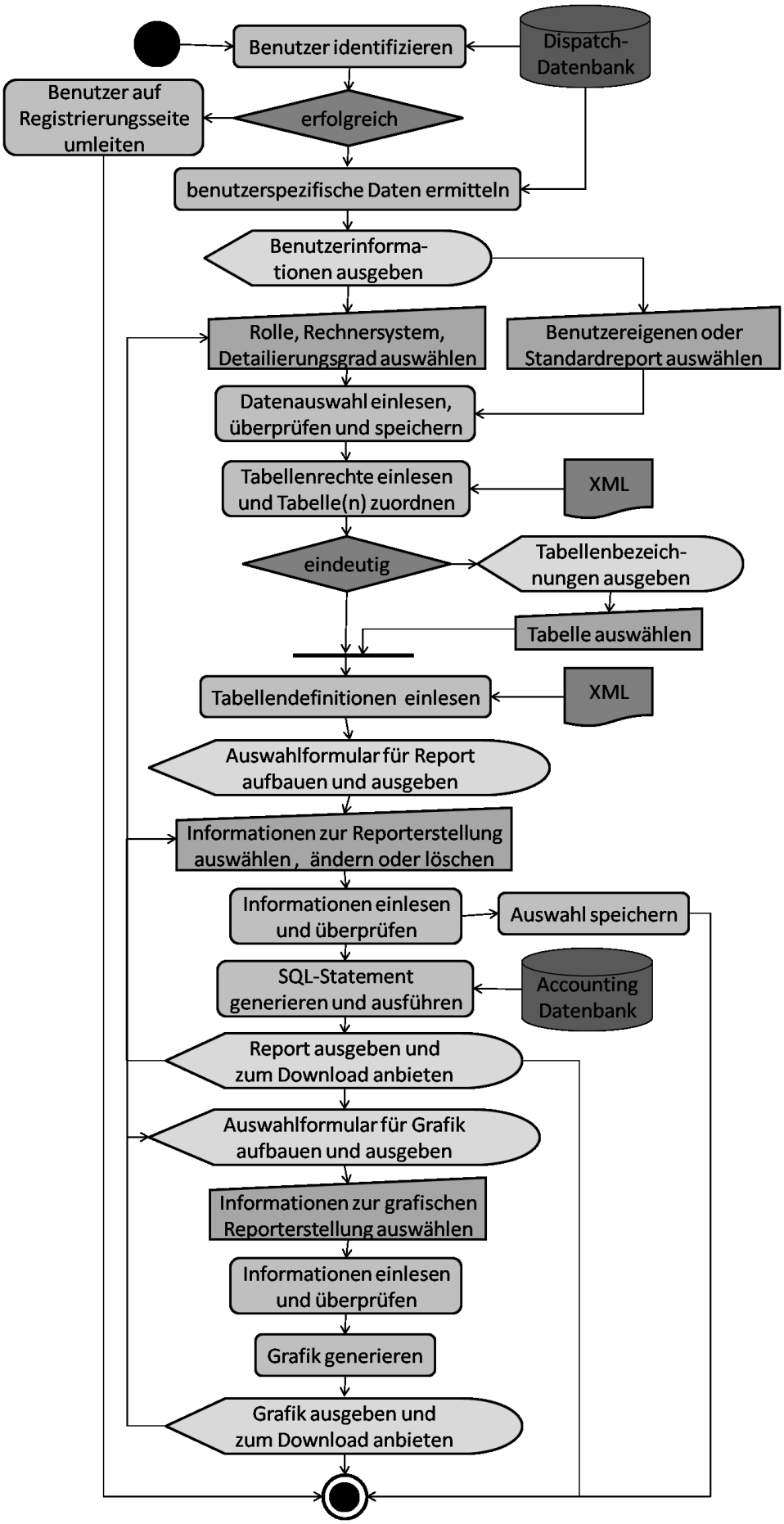

Abbildung 6-2 Ablaufdiagramm des Reporting-Systems

Mit der Abfrage des Zertifikates werden gleichzeitig die persönlichen Daten eines Benutzers gespeichert. Danach werden die zugehörigen Rollen ermittelt. Wie bereits in Abschnitt 5.2.2 beschrieben, sind die notwendigen Daten dazu nicht zentral in einer Tabelle abgelegt. Es wäre also möglich für die benötigten Daten eine zusätzliche Tabelle anzulegen. Somit erhält man die Rollen mit leichteren SQL-Abfragen. Durch diese Vorgehensweise wird jedoch unnötiger Speicherplatzbedarf generiert und eine Redundanz der Daten erzeugt. Eine andere Möglichkeit ist eine Sicht anzulegen, in der nur auf die Daten der verschiedenen Tabellen verwiesen wird. Dazu wird kein zusätzlicher Speicherplatz angelegt. Beim Zugriff wird dann der Sichtname automatisch durch seine Definition ersetzt. Dadurch werden jedoch die Abfragen zu komplex und nehmen mehr Zeit in Anspruch. Daher werden die bereits existierenden Beziehungen der Tabellen untereinander ausgenutzt und die gesamten Daten eines einzelnen Benutzers durch kombinierte Abfragen ermittelt.

Für die Erstellung der Berichte sind neben den Rollen noch weitere Daten notwendig. Dabei handelt es sich um die Projekte, die BenutzerIDs und das Rechnersystem. Über diese Daten wird gleichzeitig die Verbindung zu den Accounting-Datenbanktabellen realisiert. Für jeden Benutzer werden alle möglichen Rollen und zugleich die dazugehörigen Daten durch die kombinierten Abfragen aus der Datenbank gesucht. Damit der Zusammenhang der Daten nicht verloren geht, werden diese in einer Datenstruktur gespeichert, die in der Abbildung 6-3 dargestellt ist. Einer Rolle wie z.B. Administrator werden dafür die Projekte zugeordnet. Für ein Projekt müssen die dazugehörigen BenutzerIDs gesichert werden. Da eine BenutzerID auf mehreren Rechnersystemen gültig ist, muss sich das Reporting System auch diesen Bezug merken.

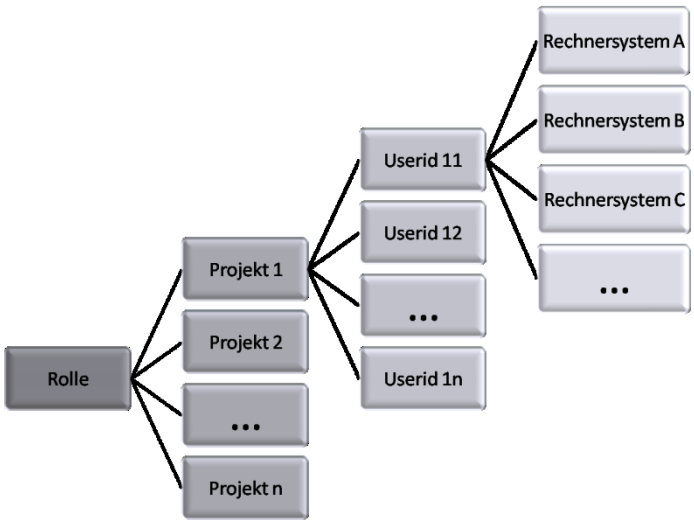

Abbildung 6-3 Datenstruktur zur Speicherung der Benutzerinformationen

Nachdem alle nötigen Daten des Benutzers aus der Dispatch-Datenbank ausgelesen wurden, werden ihm seine Daten ausgegeben. Dabei werden zunächst die persönlichen Daten des Benutzers aufgelistet. Danach folgen die ermittelten Rollen und dazugehörigen Projekte. Der Benutzer erhält somit einen rollenspezifischen Überblick. Daraus wird deutlich, welche Rollen einem Benutzer zugeordnet sind und auch welche Projekte zu einer Rolle gehören. Ein Beispiel für die Ausgabe der Benutzerinformationen ist in der Abbildung 6-4 zu sehen.

personal data

email	surname	first name	organisation	prefix number	phone number
st.meier@fz-juelich.de	Meier	Stefanie	JSC	02461/61-	1502

your roles

user		
DEISA	DGRID	JSC

project manager		
DEISA	PRACE	

project adviser		
FZJ02HIL	FZJ04HHH	FZJ05HMZ

Abbildung 6-4 Ausgabe der Benutzerinformationen

6.4.2. Reporterstellung

Anhand der Ausgabe der Benutzerdaten kann der Nutzer des Reporting-Systems zunächst eine Auswahl treffen, zu welchen Informationen er einen Bericht erstellen möchte. Durch ein Formular soll zunächst nur die gewünschte Rolle, das Rechnersystem und der Detailierungsgrad der Informationen festgelegt werden. Das Reporting-System gibt dazu die benutzerspezifischen Rollen mit den dazugehörigen Rechnersystemen aus. Diese Angaben sind in der Datenstruktur der Benutzerinformationen (Abbildung 6-3) gespeichert. An dieser Stelle soll noch kein spezielles Projekt oder eine BenutzerID ausgewählt werden, da dadurch die Flexibilität des Reports eingeschränkt wird. In der folgenden Abbildung 6-5 werden zwei personenbezogene Formulare angezeigt.

selection

Please select the role, the system and the level of detail!

role	system	level of detail
◉ user	◉ softcomp	○ combined (last 2 years)
	○ jump	◉ detailed (last 3 months)

send

selection

Please select the role, the system and the level of detail!

role	system	level of detail
○ admin	○ jump	○ combined (last 2 years)
○ adviser	○ jugene	○ detailed (last 3 months)
○ user	○ juggle	
○ manager	○ softcomp	

send

Abbildung 6-5 Auswahl der Rolle, des Rechnersystems und des Detailierungsgrades

Die Benutzerinteraktion wird, wie schon in Abschnitt 5.3.2 beschrieben, über Formulare geregelt. Die ausgewählten Daten werden dann vom Programm eingelesen und überprüft. Bei serverseitigen Anwendungen werden Benutzereingaben aus HTML-Formularen verarbeitet.

Hierbei besteht die Gefahr, dass die Eingaben nicht korrekt sind. Es kann sich dabei um ungültige Angaben handeln, die den Programmablauf manipulieren und Daten ausgeben, die eigentlich nur autorisierten Benutzern vorbehalten sind. Im schlechtesten Fall kann die Anwendung zum Absturz gebracht werden. Daher werden zunächst nach jedem Einlesen der Formulardaten die Angaben überprüft.

Da in der Datenbank mehrere Tabellen zur Verfügung stehen, muss eine Vorauswahl zu den Daten getroffen werden. Auf der Webseite können nämlich nicht alle Tabellen abgebildet werden. Dadurch würde die Übersichtlichkeit verloren gehen. Zusätzlich muss bei der Vorauswahl der Detailierungsgrad angegeben werden, weil die Daten jeweils in zwei unterschiedlich detaillierten Tabellen abgelegt sind. Falls die Informationen beider Tabellen angezeigt würden, werden dem Benutzer ähnliche Spaltenbezeichnungen angezeigt. Dadurch geht der Bezug zu den einzelnen Tabellen verloren und der Benutzer weiß in diesem Zusammenhang auch nicht, welche Inhalte er auswählen soll. Daher werden dem Benutzer immer nur die Angaben einer Datenbanktabelle ausgegeben. Die Erstellung eines Reports wird dadurch ebenfalls beschleunigt, da der Benutzer die Auswahl der Informationen schneller überblicken kann. Für die Zuordnung der Tabellen zu einer Rolle und zu einem System wird die XML-Datei `tabellenrechte.xml` eingelesen. Auszüge dieser Datei werden in der Abbildung 5-9 in Abschnitt 5.3.2 dargestellt. Die Datei wird mit Hilfe des XML-Moduls eingelesen und die Tabelle zu einer Rolle gespeichert. Durch die XML-Definition der Tabellenrechte werden einer ausgewählten Rolle, einem System und einem Detailierungsgrad eine oder mehrere Tabellen zugeordnet. Falls mehrere Tabellen zutreffen, werden dem Benutzer die Beschreibungen der Tabellen angezeigt, aus denen er sich eine Tabelle aussuchen kann. Die Beschreibungen sind eindeutig, so dass der Benutzer für den jeweiligen Verwendungszweck die richtige Tabelle findet. Danach werden die notwendigen Tabellendefinitionen aus der XML-Datei `tabellendefinitionen.xml` gelesen. Die XML-Dateien sind auf dem Server gespeichert und stehen dadurch jederzeit zur Verfügung. Die eingelesenen Informationen werden gleichzeitig für die Erstellung des Auswahlformulars und für die Kontrolle der Daten genutzt. Die Tabellenspalten sind in der XML-Datei in verschiedene Klassen eingeteilt. Dadurch verbessert sich die Übersichtlichkeit für den Benutzer und die Zusammenhänge zwischen einzelnen Tabellenspalten werden deutlich. Für die weiteren Erläuterungen wird an dieser Stelle die Definition einer Tabellenspalte in der XML-Datei in der Abbildung 6-6 gezeigt.

```
<column name="cputime" description="cpu time in sec" show="1" numeric="1"
 category="time" />
```

Abbildung 6-6 XML-Spaltendefinition

Mit Hilfe des Attributes `description` wird eine benutzerfreundliche Bezeichnung der Tabellenspalte angegeben. In den Datenbanktabellen werden alle Informationen aus einer Log-Datei gespeichert, es sind jedoch nicht alle für das Reporting notwendig. Daher wird über das Attribut `show` definiert, ob die Spalte in dem Formular überhaupt angezeigt werden soll. Durch die Möglichkeit die Daten zu gruppieren, kann ein Maximum, ein Minimum, eine Anzahl, eine Summe oder ein Mittelwert der Datensätze ermittelt werden. Die Summe und der Mittelwert können jedoch nur zu numerischen Werten gebildet werden. Durch das Attribut `numeric` wird bestimmt, ob es sich um einen numerischen Spaltenwert handelt. Das letzte Attribut `type` wird für die Angabe einer Einschränkung zur Abfrage der Daten benötigt. Dabei muss zunächst die

Kategorie der Daten unterschieden werden, da unterschiedliche Vergleichsoperatoren angeboten werden. Dies kann nicht durch die vorherigen Attribute geschlussfolgert werden und muss somit gesondert angegeben werden. Insbesondere kann diese Angabe auch für die Umrechnung in andere Einheiten verwendet werden. Eine wesentliche Bedeutung hat das Attribut auch bei der Angabe der BenutzerID und der Projekte. Dadurch wird der Zugriff auf die Daten des Benutzers und seine gewählte Rolle eingeschränkt. Wie bereits bei der Auswahl der Rolle und des Rechnersystems werden nur die Werte angezeigt, die zu einem Benutzer gehören. Dafür wird ebenfalls auf die Benutzerinformationen in der Datenstruktur zugegriffen. Diese verschiedenen Angaben müssen für die Erzeugung des Auswahlformulars bekannt sein. In der Abbildung 6-7 wird das Auswahlformular für das Linux Cluster SoftComp angezeigt.

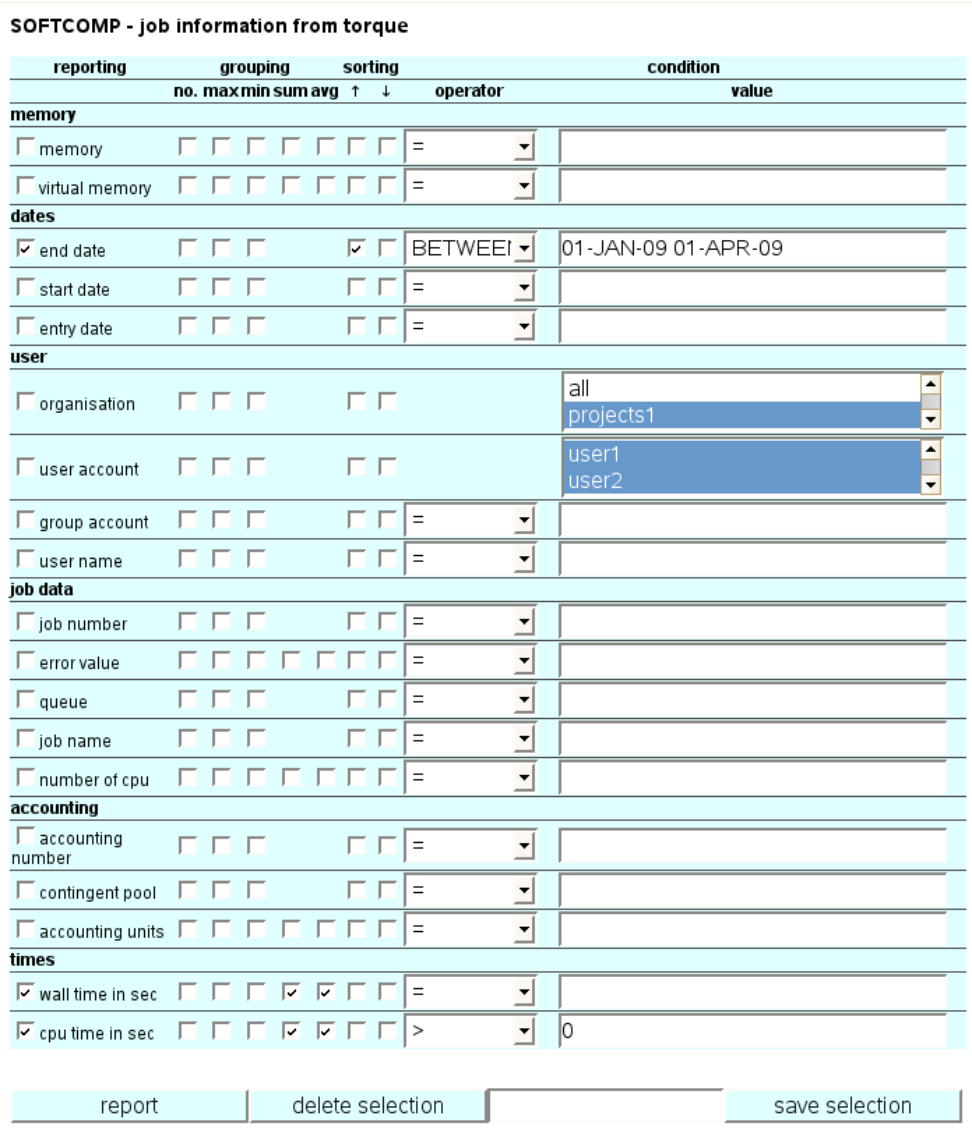

Abbildung 6-7 Auswahlformular vom Linux Cluster SoftComp

Nachdem der Benutzer eine Auswahl getroffen hat, kann er über den `report`-Button seinen Bericht generieren lassen. Dazu werden die ausgewählten Daten ebenfalls eingelesen und geprüft. Insbesondere müssen die vom Benutzer getroffenen Einschränkungen beachtet werden. Hat er keine spezielle BenutzerID ausgewählt, werden automatisch die Daten für alle ihm zugeordneten IDs generiert. Das Gleiche gilt auch für ein Projekt. Dadurch wird entsprechend der zuvor ausgewählten Rolle der Zugriff auf die zugehörigen Informationen beschränkt.

Nach der Auswahl der Daten wird das SQL-Statement generiert. Die Abfrage bezieht sich dabei immer nur auf die ausgewählte Datenbanktabelle. Mit der SQL-Abfrage wird nach den Daten in der Datenbank gesucht, d.h. die Daten werden nach den angegeben Informationen gefiltert. Die Syntax von SQL ist relativ einfach aufgebaut und semantisch an die englische Sprache angelehnt. Der Aufbau einer SQL-Abfrage (unvollständig) ist in der Abbildung 6-8 dargestellt.

```
SELECT [DISTINCT] Auswahlliste
 FROM Datenbanktabelle
 [WHERE Where-Bedingung]
 [GROUP BY (Gruppierungsattribut)+]
 [ORDER BY (Sortierungsattribut [ASC|DESC])+];
```

Abbildung 6-8 Syntax einer SQL-Abfrage

Die anzuzeigenden Felder des zu erzeugenden Reports werden in einer Auswahlliste nach dem SELECT angegeben. Sie bestimmt, welche Spalten der Datenbanktabelle auszugeben sind. Die Aufzählung der einzelnen Spaltennamen wird durch Kommas getrennt. Dabei können mit DISTINCT gleiche Datensätze ausgeblendet werden. Es wird also jeder Datensatz nur einmal ausgegeben. Durch die Auswahl von den Spalten einer Tabelle wird eine Projektion gebildet. Werden dagegen die Datensätze durch die WHERE-Bedingungen eingeschränkt, findet eine Selektion statt. Es werden nur die Daten ausgegeben, die der Bedingung entsprechen. Die Optionen für eine Selektion werden durch die Vergleichsoperatoren festgelegt. Nach dem Schlüsselwort FROM wird die Datenquelle, der Datenbank-Tabellenname angegeben. Bei der Auswahl der Daten werden sowohl die Anzeigeoptionen für die Sortierung als auch für die Aggregation aufgeführt. Nach ORDER BY werden dazu die Spaltennamen der zu sortierenden Daten und deren Sortierungsrichtung aufgezählt. ASC gibt dabei die aufsteigende und DESC die absteigende Sortierung an. Die Gruppierungsattribute werden nach dem GROUP BY aufgelistet. Dabei wird festgelegt, ob unterschiedliche Werte als einzelne Zeilen ausgegeben werden sollen oder aber die Werte der Zeile durch Addition (SUM), Durchschnitt (AVG), Minimum (MIN), Maximum (MAX) oder eine Zählung (COUNT) zu einem Ergebniswert zusammengefasst werden, der sich auf die Gruppierung bezieht. In der Auswahlliste des SELECT wird bereits bestimmt, ob die Aggregatfunktionen angewendet werden, wobei nach dem GROUP BY nur das Gruppierungsattribut angegeben wird.

Da das Formular wie eine Tabelle aufgebaut ist, können die Werte zur Erzeugung der SQL-Abfrage entsprechend gespeichert werden. In der Abbildung 6-9 wird der Tabellenkopf des Auswahlformulars mit den jeweils dazugehörigen SQL-Angaben dargestellt.

reporting	**grouping**					**sorting**	**condition**	
	no.	**max**	**min**	**sum**	**avg**	↑ ↓	**operator**	**values**
↓		↓				↓	↓	
SELECT		GROUP BY				ORDER BY	WHERE	

Abbildung 6-9 SQL-Generierung anhand der Spalten des Auswahlformulars

In der ersten Spalte des Auswahlformulars befinden sich die Bezeichnungen der einzelnen Tabellenspalten, die gleichzeitig in der Auswahlliste der SQL-Abfrage Verwendung finden. Dabei werden in dem Formular nur die Bezeichnungen der Tabelle angegeben. Für die SQL-Abfrage werden aber die Spaltennamen benötigt. Dieser Zusammenhang ist in den XML-Definitionen der Tabellen hinterlegt. In den nächsten Spalten sind die Gruppierung, die

Sortierung und die möglichen Bedingungen für eine Einschränkung jeweils zu einer Feldbezeichnung aufgelistet. Diese werden dann dem GROUP BY, dem ORDER BY oder der WHERE-Klausel zugeordnet.

Mit Hilfe der gespeicherten Werte aus dem Auswahlformular wird eine Zeichenkette generiert, die der SQL-Abfrage für die ausgewählten Informationen entspricht. Um die Daten aus den Accounting-Datenbanktabellen auszulesen, wird mit dem Datenbank-Modul eine Verbindung aufgebaut und die Abfrage ausgeführt.

Unter dem Auswahlformular befindet sich ebenfalls ein `delete`-Button, mit dem die zuvor ausgewählten Werte zurückgesetzt werden. Dafür werden die gesetzten Werte gelöscht und das Formular neu aufgebaut und ausgegeben.

Mit dem `save`-Button können die ausgewählten Informationen als benutzereigene Berichte gespeichert werden. Dazu muss in dem Textfeld vor dem Button der Name des Berichtes angegeben werden, der gleichzeitig als Name für die XML-Datei verwendet wird. Die Auswahl wird wie bei der Reporterstellung auch eingelesen und geprüft. Danach wird mit Hilfe des XML-Moduls das XML-Schema für den benutzereigenen Report in die Datei geschrieben, wie es bereits in Abschnitt 5.3.3 Standard- und benutzereigene Reports erläutert wurde. Die Daten werden so abgespeichert, dass sie den XML-Normen entsprechen. Ebenfalls müssen sie zu einem späteren Zeitpunkt den Parametern zur Erstellung eines Berichtes zugeordnet werden können. Deshalb müssen zusätzlich auch die Angaben zur Auswahl einer Datenbanktabelle gespeichert werden. Beim erneuten Aufruf des Programmes werden dem Benutzer neben den Standard-Reports auch die gespeicherten benutzereigenen Reports zur Auswahl angeboten. Sucht sich der Benutzer einen solchen Report aus, wird zunächst die XML-Definition aus der ausgewählten Datei eingelesen. Danach werden die Parameter zur Auswahl der Datenquelle gesetzt und das Auswahlformular mit den gespeicherten Werten angezeigt. Der Benutzer kann sich nun die Daten zu seinem gespeicherten Bericht generieren lassen oder die Auswahl nach seinen eigenen Wünschen verändern. Für die Standard- und benutzereigenen Reports werden die Funktionen zur Erstellung der flexiblen Reports mit bereits vordefinierten Werten genutzt.

6.4.3. Reportdarstellung

Die aus der Datenbankabfrage resultierenden Daten werden als Tabelle oder Liste auf der Webseite des Reporting Tools ausgegeben. Für diese Ausgabe wird ebenfalls das CGI-Modul verwendet. Das Modul bietet dazu einige HTML-Hilfsfunktionen an. Die Anwendung ist besonders nützlich, da das Ergebnis der Datenbankabfrage formatiert werden soll. Für die Tabelle oder Liste wird der HTML-Code mit dem dazugehörigen Dateninhalt erzeugt und ausgegeben. Durch die Verwendung von HTML werden die Daten in der Tabelle statisch ausgegeben, dadurch muss für eine andere Sortierung oder Gruppierung der Report neu erzeugt werden. Bei der Ausgabe wird auch darauf geachtet, dass nicht zu viele Zeilen einer Tabelle angezeigt werden. Die Anzahl der Zeilen wird also beschränkt. Die Anzeige der gesamten Tabelle wird jedoch ebenso ermöglicht. Falls ein Benutzer alle Daten aufgelistet haben möchte, muss er nur auf den aufgeführten *More*-Button klicken.

Wie bereits in Abschnitt 5.4.1 erwähnt, werden die Berichte für jeden Benutzer speziell abgelegt. Dazu wird ein benutzereigener Ordner auf dem Webserver angelegt, in dem seine Dateien gespeichert werden.

Nachdem die Daten des zuvor erstellten Reports angezeigt werden, kann sich der Benutzer über ein Formular das gewünschte Diagramm zusammenstellen. In der Abbildung 6-10 wird dieses Formular dargestellt.

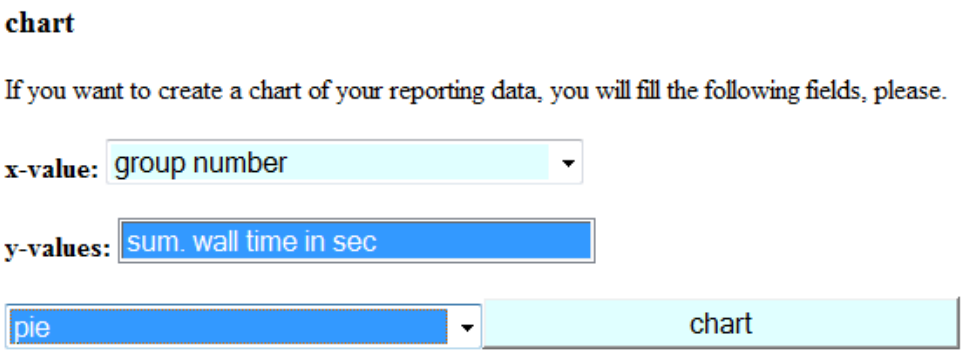

Abbildung 6-10 Formular für das Erstellen eines Diagramms

Für das Erzeugen eines Diagramms müssen numerische Werte vorhanden sein, ansonsten ist keine Darstellung möglich. In diesem Fall wird das Formular nicht ausgegeben. Andernfalls kann der Benutzer nach seinen Wünschen den x-Wert, den y-Wert und den Diagrammtyp auswählen. Um dem Benutzer die Möglichkeit zu geben, verschiedene Datenreihen in einem Diagramm anzuzeigen, kann er auch mehrere y-Werte markieren. Dies ist jedoch nur für Säulen- und Liniendiagramme möglich, da für ein Kreisdiagramm mehrere Datenreihen nicht sinnvoll sind.

Nachdem die Auswahl des Benutzers vom Programm eingelesen und überprüft wurde, kann nun das gewünschte Diagramm erstellt werden. Dazu wird die Grafik dynamisch erzeugt, da die Daten für ein Diagramm auf veränderlichen Daten eines Reports beruhen. Somit wird eine grafische Repräsentation der ausgewählten Daten erstellt.

Für die Anzeige des Diagramms wird vorher getestet, ob der Browser SVG anzeigen kann. Der Test des Browser auf SVG-Unterstützung basiert auf JavaScript und VBScript. Im Browser des Benutzers darf dazu das Ausführen von Skripten nicht deaktiviert werden. Das Skript überprüft dabei, ob das SVG-Plug-In vorhanden ist oder der Browser den *MIME*-Typ `image/svg+xml` (*Multipurpose Internet Mail Extension*) akzeptiert. Ist dies der Fall, wird die Anzeige von SVG-Grafiken vom System unterstützt. SVG-Grafiken können wie andere multimediale Objekte in HTML eingebettet werden. Falls ein Browser kein SVG-Plug-In besitzt, war die Überprüfung negativ. In diesem Fall wird ein Alternativtext ausgegeben.

Anschließend wird die SVG-Datei erstellt und ebenfalls in dem Benutzer-Ordner abgelegt. Für die vom Benutzer ausgewählten Datenreihen werden die zwischengespeicherten Daten aus der Datenbank benutzt. Die Reportdaten werden dann an das Grafik-Modul übergeben. Je nach Diagrammtyp wird ein Graph-Objekt mit den dazugehörigen Datensätzen erzeugt. Dieses Objekt enthält dann alle notwendigen Daten zur Erstellung der skalierbaren Vektorgrafik u.a. die Höhe und Breite der Grafik, die Beschriftungen der Achsen und der Datenpunkte und die Legende. Nachdem die XML-Beschreibung der Grafik ausgegeben wurde, wird die Datei in HTML eingebunden. Dadurch kann sie auf der Webseite des Reporting-Systems angezeigt werden.

Außerdem werden die erzeugten Dateien dem Benutzer zur Verfügung gestellt. Ihm wird somit die Möglichkeit gegeben, die erstellten Reports zu speichern und später auch offline zu

verwenden. Die in dem Benutzer-Ordner angelegten Dateien werden in der Abbildung 6-11 angezeigt.

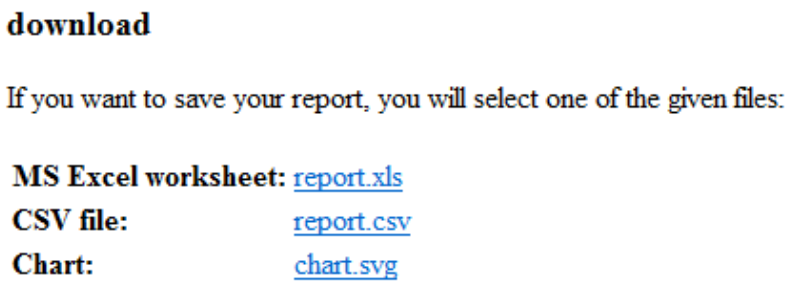

Abbildung 6-11 Formular zum Download der Reports

Kapitel 7 Zusammenfassung und Ausblick

Das Reporting ist ein wichtiges Instrument im JSC, um relevante Informationen zu dem Betrieb und den genutzten Ressourcen der Rechnersysteme zu erhalten. Dabei sollen die Berichte die notwendigen Daten auch für zukünftige Entscheidungen z.B. zur Planung der Rechenzeitverteilung bereitstellen. Zuvor wurden für verschiedene Anforderungen der jeweiligen Benutzergruppen unterschiedliche Vorgehensweisen zur Erstellung der Berichte angewandt. Ziel dieser Diplomarbeit war deshalb die Entwicklung eines einheitlichen und rollenbasierten Reporting-Systems für den Einsatz im JSC.

Die in der Arbeit vorgestellten Softwaresysteme waren hierfür jedoch nicht geeignet, da sie den ausgearbeiteten Anforderungen und Zielen wie z.B. eine einfache und flexible Erstellung eines Berichtes nicht gerecht wurden. Demzufolge wurde ein Konzept für ein neues Reporting-System im JSC entwickelt und durch eine Implementierung getestet.

Um einen einheitlichen und zentralen Zugang für jeden Benutzer zu schaffen, wurde als zugrundeliegende Technik eine Webanwendung gewählt. Somit kann das Reporting-System von jedem beliebigen Rechner mit Internetzugang benutzt werden. Es ist also ein standortunabhängiger und zeitnaher Zugriff auf die Informationen möglich. Für die Sicherheit wird die Kommunikation zwischen dem Benutzer und dem System basierend auf Zertifikaten geregelt. Sie dienen gleichzeitig auch zur Identifizierung und gegenseitigen Authentifizierung. In Zukunft ist für die Benutzer, die kein Zertifikat besitzen, ein weiterer Zugang z.B. Passwörter angedacht. Hierfür müssen ebenfalls die erarbeiteten Sicherheitsaspekte berücksichtigt werden.

Da alle Benutzer der Supercomputer und Server-Systeme das Reporting-System benutzen können, wurde ein Berechtigungskonzept mit verschiedenen Rollen entwickelt. Dadurch wird sichergestellt, dass jeder Benutzer nur Einblick in seine durch die Rolle beschränkten Daten erhält. Zurzeit werden die Rolle des Nutzers, des Leiters und des Betreuers von einem Projekt sowie die Rolle des Administrators vom Reporting-System angeboten. Für eine Erweiterung des Rollenkonzeptes wäre es denkbar, z.B. die Rolle des Systemadministrators eines Rechnersystems noch hinzuzufügen.

Für die Erstellung der Reports stehen dem Benutzer Standardberichte zur Verfügung, die zunächst den allgemeinen Bedarf an Informationen abdecken. Weiterhin ist es aber auch möglich die Berichte selbst nach eigenen Wünschen zusammenzustellen und als benutzereigene Reports zu speichern. Dem Benutzer wird dadurch eine flexible und variable Art der Reporterstellung ermöglicht und nicht nur vordefinierte Berichte angeboten. Dies stellt eine wesentliche Verbesserung gegenüber der bestehenden Vorgehensweise dar. Als Datenbasis für die Berichte dienen die Accounting-Datenbanktabellen. Durch den Zugriff auf diese Daten wird die Integrität gewährleistet und die erstellten Berichte sind inhaltlich korrekt. Durch ein XML-Schema werden die Daten definiert und somit eine Sicht auf die Datenbank ermöglicht. Dadurch erfolgt eine Beschreibung der Tabellen, so dass jederzeit neue Tabellen wie z.B. bei der Anschaffung neuer Rechnersysteme hinzugefügt werden können. Die Auswahl der gewünschten Informationen wird durch einheitliche Bezeichnungen vereinfacht. Zudem wird dem Benutzer durch die Verwendung von Formularen eine Hilfestellung bei der Erstellung der Berichte gegeben. Die bisherigen Tests zum Umgang mit der Weboberfläche zeigen, dass sie intuitiv und einfach zu bedienen ist. Dabei ist sie trotz der umfangreichen Funktionalität und Auswahl der Daten übersichtlich gestaltet.

Die Darstellung der Daten zu den ausgewählten Informationen wird durch die Verwendung von Tabellen und Listen klar strukturiert. Darüberhinaus können die Berichte auch in grafischer Form als Linien-, Säulen- oder Kreisdiagramm angezeigt werden. Sie werden in dem vektorgrafischen Format SVG dynamisch zur Laufzeit erzeugt. Mit dem neuen Reporting-System haben nun alle Benutzergruppen die Möglichkeit ihre Daten als Grafiken darzustellen. Dies war mit dem alten System nicht möglich. Die verschiedenen Ausgabemöglichkeiten der Berichte lassen sich künftig durch andere Diagrammtypen erweitern. Denkbar wären z.B. gestapelte Säulendiagramme oder unterteilte Flächendiagramme. Bei diesen Säulendiagrammen wird die Aufmerksamkeit auf die Zusammensetzung und die Veränderung der verschiedenen Komponenten gerichtet. Die Flächendiagramme hingegen können die Entwicklungen von mehreren Datenmengen aufzeigen. Für eine genauere Darstellung des Sachverhaltes wären sie jedoch weniger geeignet.

Die erzeugten Berichte werden zum Download angeboten, so dass sie vom Benutzer selbst heruntergeladen werden können. Sie werden dabei in gebräuchlichen Formaten wie XLS angeboten, sodass sie auch von anderen Programmen genutzt werden können. Für den weiteren Ausbau des Reporting-Systems könnten die Berichte nach der Erstellung auch per Email versendet werden.

Als Erweiterung des Informationsangebotes kann zusätzlich zur Ausgabe der IST-Daten eine Kalkulation eingeführt werden. Zu den Daten der Berichte lassen sich ergänzend Vergleichswerte für Trends und Analysen berechnen. Dazu können die Auswertungs- und Analysefunktionen erweitert werden.

Weitere Arbeiten könnten die Effizienz der Webanwendung optimieren, indem das CGI-Programm als persistenter Prozess ausgeführt wird. Dazu wird nach der Bearbeitung der Anfrage der Prozess nicht beendet, sondern steht für weitere Anwendungen zur Verfügung. So erzielt man eine höhere Performance und nutzt die Vorteile der Prozessisolation. Eine Realisierung davon ist FastCGI. Hierfür ist ein eigenes Protokoll zwischen dem Web-Server und der Anwendung erforderlich. Eine zweite Technologie ist mod_perl. Das Modul dient der Erweiterung des Apache-Webservers und integriert den Perl-Interpreter in den Webserver.

Die hier vorgeschlagenen Erweiterungen und Anpassungen sind durch das modulare Software-Design des Reporting-Systems relativ einfach zu integrieren. Das System wurde zunächst auf einem Testserver implementiert. Nach ausführlichen Tests sollen die bisherigen Verfahren zur Reporterstellung abgelöst werden. In den kommenden Wochen wird für den praktischen Einsatz das Reporting-System auf den Webserver des JSC portiert. Dies wird auch noch einige Anpassungen nach sich ziehen. In einer Einführungsphase sollen daher auftretende Probleme erkannt und behoben werden und eventuell neue Benutzerwünsche (z.B. bei der Bedienung der Webanwendung) implementiert werden. Bei der Entwicklung wurden bereits mögliche Erweiterungen und Neuerungen berücksichtigt, sodass diese in Zukunft ohne Probleme nachträglich integriert werden können.

Mit dem entwickelten Reporting-System wurden ein einheitlicher und sicherer Zugang und eine rollenbasierte Erstellung der Berichte verwirklicht. Es bietet zum einen Standardberichte an und ermöglicht zum anderen auch das flexible und variable Zusammenstellen der Berichte.

Anhang

Top-User-Listen

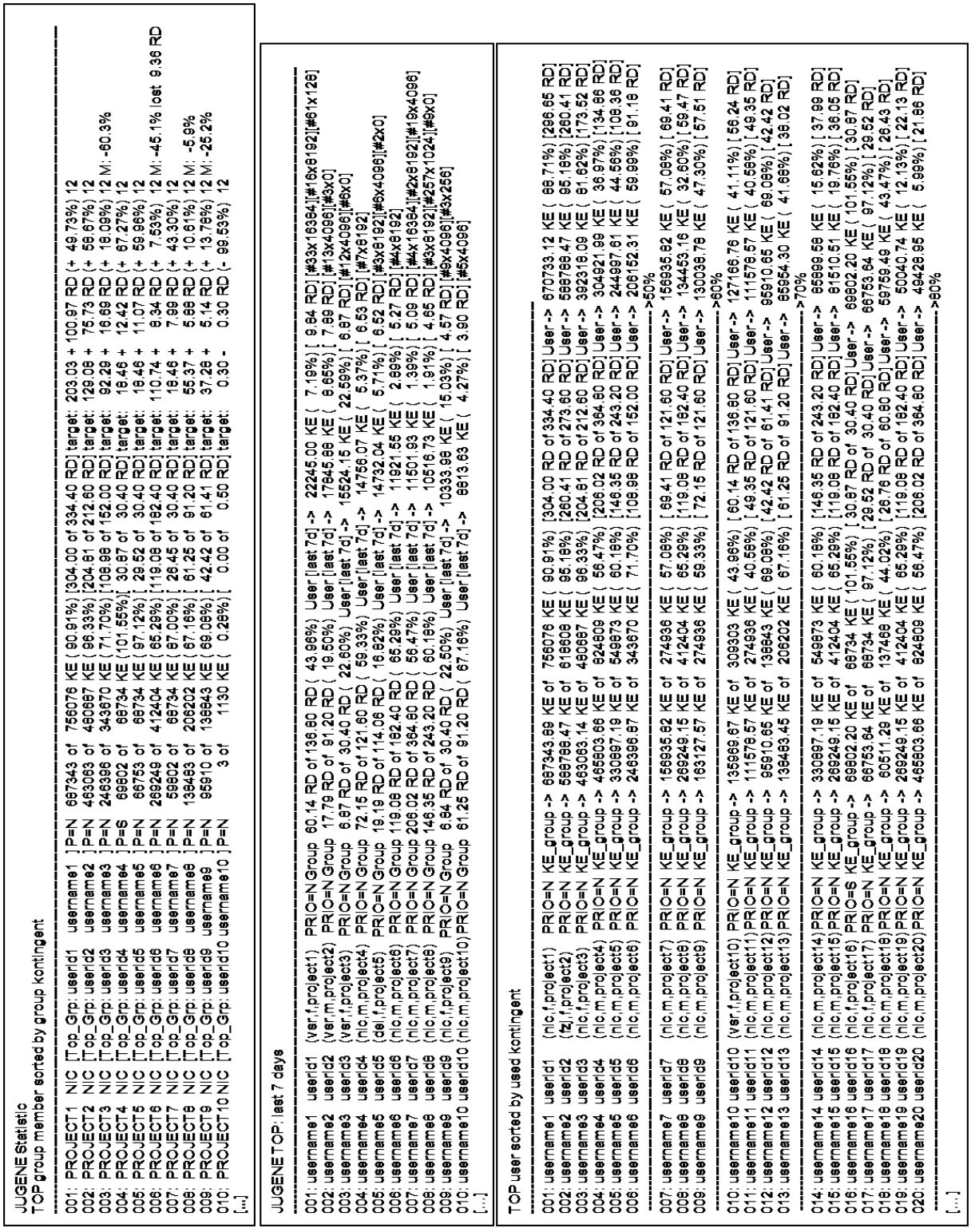

Abbildung A-1 Top-User-Listen des Supercomputers JUGENE

Benutzerzertifikat

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 222014886 (0xd3bada6)
        Signature Algorithm: sha1WithRSAEncryption
        Issuer: C=DE, O=Forschungszentrum Juelich GmbH, CN=FZJ   Certification Authority -
                    G02/emailAddress=ca@fz-juelich.de
        Validity
            Not Before: Nov 13 13:48:07 2008 GMT
            Not After : Nov 13 13:48:07 2011 GMT
        Subject: C=DE, O=Forschungszentrum Juelich GmbH, OU=JSC, CN=Cornelia Geyer
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (2048 bit)
                Modulus (2048 bit):
                    00:e5:82:c2:72:bd:2d:2b:5a:39:45:3c:86:2c:80:
                    55:49:2b:b1:9c:62:74:7f:57:34:57:21:f9:a4:47:
                    6d:e3:29:f7:b4:ff:b3:aa:e1:23:8c:60:dc:ba:7d:
                    41:5d:f5:ca:62:f4:72:5f:d8:9c:ba:b2:2d:f2:37:
                    64:ba:d3:37:87:c1:19:ea:91:9e:8e:4f:62:ab:56:
                    9c:ab:21:40:3a:c9:0f:9d:bd:e0:92:d6:ff:62:1b:
                    c3:5a:83:b5:4d:7a:68:3f:18:cf:1b:97:b1:a2:86:
                    b4:24:05:2e:5d:31:63:d2:cc:bb:2d:3c:e7:ae:2c:
                    36:3b:5c:6b:65:d5:82:ac:3f:b0:01:7e:83:07:38:
                    7f:f6:64:9d:77:22:e3:a4:00:e0:d7:cd:db:14:73:
                    43:bf:7d:fc:7c:56:5e:94:d2:05:9b:42:91:36:64:
                    62:b4:9a:17:99:97:c0:91:02:9c:1c:4b:88:5a:19:
                    e9:7c:0b:a2:0c:c9:78:ea:b9:26:2b:ce:18:ea:e1:
                    e2:b9:4d:50:42:be:69:a3:f5:fb:f8:ec:3a:ef:da:
                    27:50:9f:aa:6e:eb:43:5f:5d:5b:e7:a8:01:05:b1:
                    b0:4d:83:46:89:e5:a3:d8:83:d4:af:b0:89:10:6b:
                    c3:28:da:75:d4:90:b7:ef:54:64:8e:98:32:d2:a2:
                    ca:f1
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Key Usage:
                Digital Signature, Non Repudiation, Key Encipherment
            X509v3 Extended Key Usage:
                TLS Web Client Authentication, E-mail Protection
            X509v3 Subject Key Identifier:
                00:05:BD:22:E4:70:71:E7:FD:3D:A6:86:CD:E2:BA:4C:17:3A:7D:C9
            X509v3 Authority Key Identifier:
                keyid: 20:BD:71:89:A5:C9:2A:33:2F:11:E0:1F:E5:94:26:56:0F:
                                11:02:25
            X509v3 Subject Alternative Name:
                email:c.geyer@fz-juelich.de
            X509v3 CRL Distribution Points:
                URI:http://cdp1.pca.dfn.de/fzj-ca/pub/crl/g_cacrl.crl
                URI:http://cdp2.pca.dfn.de/fzj-ca/pub/crl/g_cacrl.crl
            Authority Information Access:
                CA Issuers - URI:http://cdp1.pca.dfn.de/fzj-ca/pub/cacert/g_cacert.crt
                CA Issuers - URI:http://cdp2.pca.dfn.de/fzj-ca/pub/cacert/g_cacert.crt
    Signature Algorithm: sha1WithRSAEncryption
        0f:77:ac:e3:67:77:d5:62:46:0e:50:91:eb:1d:fc:5c:e5:0b:
        22:4e:83:39:4c:58:ba:0e:07:f3:3f:e3:a0:54:71:25:66:a8:
        3b:49:8e:63:76:b0:90:e5:f6:93:35:ac:89:f7:fc:79:f0:14:
        03:08:bb:76:21:80:27:ee:2a:4c:b0:57:7b:66:6f:49:d4:01:
        61:ac:03:53:7a:6e:50:63:a5:a5:bd:e6:ed:e2:a1:d5:a4:c5:
        b0:d6:9e:fe:38:98:f2:d6:3b:0e:af:d2:1c:f3:9a:92:a6:cf:
        c3:52:63:22:03:d1:1b:49:3a:f9:dc:cd:23:1a:4f:1e:54:c4:
        e0:76:ec:44:8d:31:16:b3:a6:b4:d6:b8:b0:9f:6f:59:3b:e0:
        b7:4d:b6:d5:be:27:1d:87:7c:41:49:f3:8f:37:82:a0:8a:f4:
        47:d0:c0:00:b4:83:b4:20:13:a2:76:a5:ed:bd:87:cd:b0:83:
        a0:e7:fb:21:38:50:44:09:6a:6d:a5:fb:22:76:cc:a4:22:51:
        38:0d:55:2b:92:40:da:8c:9e:6c:42:cf:db:c1:fc:63:92:7e:
        6c:bd:7c:29:45:10:3c:ca:66:66:a3:29:79:1f:a0:6e:ec:46:
        ab:a6:14:7b:07:f4:49:b8:01:1c:37:fa:31:f5:d0:8b:3c:79:
        2c:d7:12:56
```

Abbildung A-2 Text-Darstellung eines digitalen Benutzer-Zertifikats

Literaturverzeichnis

[1] Bos, B. & Fischer, H. (13. November 2001). *XML in 10 Punkten*. Abgerufen am 25. März 2009 von http://www.w3c.de/Misc/XML-in-10-points.html

[2] Doumack, A. (Oktober 2008). *Evaluierung von Reporting-Frameworks am Beispiel der ausgewählten Open-Source-Anwendungen.* Abgerufen am 20. 1 2009 von http://faeskorn-woyke.de/wp-content/uploads/diplom_eval_ado.pdf

[3] Eclipse Foundation. (2009). *BIRT*. Abgerufen am 2. Februar 2009 von http://www.eclipse.org/birt/phoenix/

[4] Federrath, H. (24. August 2008). *Sicherheit im einzelnen Rechner.* Abgerufen am 9. März 2009 von http://www-sec.uni-regensburg.de/security/folien/40ReSi.pdf

[5] Finger, R. (Februar 2006). *BI-Betriebsmodelle auf dem Prüfstand.* Abgerufen am 16. Februar 2009 von http://www.tdwi.eu/tdwi-wissen/download.html?tx_fhkb_pi1%5Bitemid%5D=198&tx_fhkb_pi1%

[6] Gluchowski, P., Gabriel, R. & Dittmar, C. (2008). *Management Support Systeme und Business Intelligence. Computergestützte Informationssysteme für Fach- und Führungskräfte.* Berlin Heidelberg: Springer Verlag 2. Auflage S. 90-93, 108-109, 205-220.

[7] Guelich, S., Gundavaram, S. & Birznieks, G. (2001). *CGI Programmierung mit Perl.* Köln: O'Reilly Verlag, 2. Auflage.

[8] Hauke, R. (2008). *Web Programmierung.* Abgerufen am 26. Februar 2009 von http://www.ku-eichstaett.de/Fakultaeten/WWF/Lehrstuehle/WI/Lehre/web_prog/

[9] Held, M. & Klose, I. (10. Dezember 2007). *Business Intelligence mit Pentaho.* Abgerufen am 2. Februar 2009 von http://www.heise.de/open/Business-Intelligence-mit-Pentaho--/artikel/98599

[10] JasperSoft. (2009). *JasperReports.* Abgerufen am 2. Februar 2009 von http://www.jaspersoft.com/JasperSoft_JasperReports.html

[11] Jung, H. (2007). *Controlling.* München: Oldenbourg Wissenschaftsverlag 2.Auflage S. 141-143.

[12] Kalenborn, A. (2008). *Grundlagen von Webanwendungen.* Abgerufen am 26. Februar 2009 von http://www.uni-trier.de/fileadmin/fb4/prof/INF/WI1/Lehrmaterialien/WI/Content_Management/02_-_Grundlagen_von_Webanwendungen.pdf

[13] Kappel, G., Pröll, B., Reich, S. & Retschitzegger, W. (2004). *Web Engineering, Systematische Entwicklung von Webanwendungen.* Heidelberg: dpunkt-Verlag S.4.

[14] Kersken, S. (2006). *Apache 2*. Bonn: Galileo Press, 2. Auflage S. 478 - 515.

[15] Kleijn, A. (8. 6 2006). *Business Intelligence mit Open Source*. Abgerufen am 2. 2 2009 von http://www.heise.de/open/Business-Intelligence-mit-Open-Source--/artikel/73725

[16] Neumann, A. & Winter, A. M. (2001). *SVG - Ein zukünftiger Eckstein der WWW-Infrastruktur*. Abgerufen am 25. März 2009 von http://www.carto.net/papers/svg/articles/paper_ugra_zurich_2001.pdf

[17] Pentaho Open Source Business Intelligence. (2009). *Pentaho Reporting.* Abgerufen am 2. Februar 2009 von http://www.pentaho.com/docs/pentaho_reporting.pdf

[18] Pfitzmann, A. (15. Oktober 2000). *Datensicherheit und Kryptographie.* Abgerufen am 9. März 2009 von http://dud.inf.tu-dresden.de/~pfitza/DSuKrypt.pdf

[19] Richter, H. (14. Juni 2005). *Verschlüsselung, digitale Signaturen, Zertifikate*. Abgerufen am 9. März 2009 von http://www.lrz-muenchen.de/services/pki/einf/einf-3.html

[20] Schoenebeck, F. J. & Sczimarowsky, M. (7. Mai 2002). *JuNet/Internet - Einsatz von X.509-Zertifikaten*. Abgerufen am 9. März 2009 von http://www.fz-juelich.de/jsc/docs/tki/tki_html/t0365/t0365.html

[21] SoftEd Systems. (2009). *Wie funktioniert HTTPS?* Abgerufen am 9. März 2009 von http://www.softed.de/fachthema/https.aspx

[22] Todt, M. & Bauer, P. (Oktober 2004). *Benutzer-Authentifizierung.* Abgerufen am 9. März 2009 von http://www.bacher.at/download/loesungen/bacher.at_benutzer-authentifizierung.pdf

[23] Turau, V. (1999). *Techniken zur Realisierung Web-basierter Anwendungen.* Springer Verlag Informatik Spektrum 22 Heft 3 S.3 - 12.

[24] Turowski, K. (2008). *Web Engineering.* Abgerufen am 26. Februar 2009 von http://www.wi2.info/de/web-engineering/index.php

[25] Ullenboom, C. (2009). *Java ist auch eine Insel - Das Model-View-Controller-Konzept*. Abgerufen am 7. April 2009 von http://openbook.galileocomputing.de/javainsel8/javainsel_16_015.htm

[26] Wikipedia, Die freie Enzyklopädie. (4. März 2009). *Einmalkennwort*. Abgerufen am 9. März 2009 von http://de.wikipedia.org/wiki/Einmalkennwort

[27] Wikipedia, Die freie Enzyklopädie. (9. Oktober 2007). *Reportgenerator*. Abgerufen am 2. Februar 2009 von http://de.wikipedia.org/wiki/Reportgenerator